МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕРСИТЕТ
Факультет кібербезпеки, комп'ютерної та програмної інженерії
Кафедра комп'ютерних інформаційних технологій

ДОПУСТИТИ ДО ЗАХИСТУ

Завідувач кафедри

Савченко А.С.

«___»_____2020 р.

ДИПЛОМНА РОБОТА
(ПОЯСНЮВАЛЬНА ЗАПИСКА)

ВИПУСКНИКА ОСВІТНЬОГО СТУПЕНЯ
"МАГІСТРА"

ЗА СПЕЦІАЛІЗАЦІЄЮ "ІНФОРМАЦІЙНІ УПРАВЛЯЮЧІ СИСТЕМИ ТА ТЕХНОЛОГІЇ (ЗА ГАЛУЗЯМИ)"

Тема: «Домашнє портативне хмарне сховище»

Виконавець:: Рознай Олександр Іванович

Керівник: професор Воронін Альберт Миколайович

Нормоконтролер: Райчев І.Е.

Київ 2020

НАЦІОНАЛЬНИЙ АВІАЦІЙНИЙ УНІВЕСИТЕТ

Факультет кібербезпеки, комп’ютерної та програмної інженерії

Кафедра комп’ютерних інформаційних технологій

Спеціальність 122 “Комп’ютерні науки та інформаційні технології”

Спеціалізація «Інформаційні управляючі системи та технології (за галузями)»

ЗАТВЕРДЖУЮ

Завідувач кафедри

Савченко А.С.

“ ” _____2019р.

З А В Д А Н Н Я

на виконання дипломної роботи студента

Розная Олександра Івановича
(прізвище, ім’я, по батькові)

1. Тема проекту (роботи): Домашнє портативне хмарне сховище» затверджена наказом ректора №2175/ст. від 25.09.2019р..
2. Термін виконання проекту (роботи): з 26.09.2019р. по 03.02.2020р.
3. Вихідні данні до проекту (роботи): хмарне сховище, Raspberry Pi міні-ПК
4. Зміст пояснювальної записки (перелік питань, що підлягають розробці): вступ, аналітичний огляд і постановка завдання, висновок.
5. Перелік обов’язкового графічного матеріалу: інтерфейс хмарного сховища, термінал операційної системи Raspbian, схеми роботи технологій

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Завдання	Термін виконання	Підпис керівника
1.	Аналіз літератури та джерел за темою дипломного проекту.	14.10.19р.– 20.10.19	
2.	Розроблення та затвердження плану дипломного проекту.	21.10.19– 22.10.19	
3.	Проведення консультації з науковим керівником щодо створення першого розділу.	23.10.19 – 27.10.19	
4.	Розробка розділу 1	30.10.19 – 22.11.19	
5.	Розробка розділу 2	23.11.19 – 08.12.19	
6.	Розробка розділу 3	09.12.19 – 22.12.19	
7.	Висновки та оформлення пояснювальної записки дипломного проекту.	25.12.19 – 29.12.19	
8.	Підписання необхідних документів у встановленому порядку.	15.01.20-19.01.20	
9.	Підготовка до захисту та попередній захист дипломного проекту на випусковій кафедрі дипломного проекту	22.01.20 – 31.01.20	

7. Дата видачі завдання: 14.10.2019р.

Керівник дипломного проекту _____ Воронін А.М.

(підпис керівника) (П.І.Б.)

Завдання прийняв до виконання _____ Рознай О.І.

(підпис випускника) (П.І.Б.)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту роботи «Домашнє портативне хмарне сховище» викладена на 74с., містить 14 рис., табл.1, 9 літературних джерел.

Ключові слова: Cloud, Raspberry Pi, Linux, Terminal, OwnCloud, NAS, Хмарні технології , Raspbian, SSH, API, Open Source, PHP, SSL, OpenSSL, IoT.

***Об'єкт дослідження*:** Технологія хмарного зберігання файлів з використанням міні комп'ютера Raspberry Pi та відкритого програмного забезпечення OwnCloud.

Предмет дослідження: Домашнє портативне хмарне сховище

Мета роботи: Розробите особисте домашнє хмарне сховище на базі програмного забезпечення ownCloud з відкритим кодом та міні-ПК Raspberry Pi.

Методи дослідження: Використання ownCloud для створення мережевого сховища файлів на Raspberry Pi на базі програмних компонентів веб-сервера.

Отримані результати: Було розроблене домашнє портативне хмарне сховище, яке стабільно працює і досі.

Результати дипломної роботи використовуються в домашніх умовах та можуть бути використанні і технологічній сфері діяльності.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

БД – база даних;

ОС - операційна система;

ПЗ - програмне забезпечення;

ПК - персональний комп’ютер;

RPi – система захисту інформації

SSL – сертифікат безпеки;

DNS - Domain Name System;

SSH – Secure SHell;

NTFS - New Technology File System;

IoT – Internet of Things.

ВСТУП

Важко знайти людину, яка не носить з собою флешку. Чи працюєте ви в компанії або вчіться в університеті, вам потрібен цей маленький інструмент кожен день. Але розумієш ти, скільки проблем у тебе з флешкою?

По-перше, її легко втратити. Деякі люди змінюють флешки більше, ніж шкарпетки та рукавички, тому що вони залишають їх всюди. Максимальна місткість флешок складає 128 ГБ (цього буде достатньо, щоб завантажити всі сезони гри престолів у відмінній якості). Що за проблема? Що ж, пристрій з такою ємністю коштує дорого, і якщо ви його втратите, ви напевно пошкодуєте.

Але з хмарними накопичувачами ви можете врятувати свою освіту від проблем, пов'язаних з флешками. Це стане і дивним об'єктом, який ви не пошкодуєте втратити.

- Оптимізуйте вашу співпрацю. Хмарні накопичувачі - це ідеальний інструмент для негайного обміну даними.
- Створення резервної копії ваших особистих файлів. Зазвичай, коли комп'ютер або смартфон виходять з ладу, власники цих пристроїв спочатку страждають через втрачених грошей, а потім через втрату даних.
- Захистіть свою роботу від втрати. Кожна інтелектуальна робота дорогоцінна. Але в цифровому світі його можна знищити за кілька хвилин. Є багато випадків, коли студенти Google «допомагають мені написати есе» в пошуковій записи, тому що вони не зберегли свій файл з домашньою роботою і витрачають час даремно.
- Отримайте більше місця за менші гроші. Плата за необмежену зберігання в хмарі дешевше, ніж покупка і обслуговування великої кількості місця на жорсткому диску. Люди як і раніше купують жорсткі диски на декількох рівнях зберігання в своїх будинках і офісах. Але будь-яка фізична пристрій ви можете упустити з різних причин.

РОЗДІЛ 1

ПОНЯТТЯ ХМАРНИХ СХОВИЩ ТА ЇХ ПРИЗНАЧЕННЯ

1.1 Загальне визначення хмарних сховищ

Хмарне сховище - це модель зберігання комп'ютерних даних, в якій цифрові дані зберігаються в логічних пулах. Фізичне сховище охоплює кілька серверів (іноді в декількох місцях), а фізичне середовище зазвичай належить і управляється хостингової компанією. Ці постачальники хмарних сховищ несуть відповідальність за забезпечення доступності та доступності даних, а також за захист і роботу фізичної середовища. Люди і організації купують або орендують сховища у провайдерів для зберігання даних користувачів, організацій або додатків.

Доступ до хмарних сховищ може здійснюватися через спільно розміщену хмарну обчислювальну службу, інтерфейс прикладного програмування (API) веб-служби або функції, які залежать API, такі як хмарне настільне сховище, шлюз хмарного сховища або системи управління контентом на основі Web.

Тобто, хмарне сховище:

1. Складається з багатьох розподілених ресурсів, але все ще діє як об'єднана, чи то в об'єднаній архітектурі хмари зберігання даних

2. Висока відмовостійкість завдяки надмірності і розподілу даних

3. Висока надійність завдяки створенню версійних копій.

4. Як правило, в кінцевому підсумку відповідає реплікам даних.

Нижче ми порівняємо з вами 4 найпопулярніші хмарні сервіси для зберігання даних від найвідоміших IT-гігантів. Але чи безпечно віддавати свої дані іншим компаніям? Адже, вони можуть використовувати їх у своїх цілях, наприклад, для кращої пошукової видачі та для тергетованої реклами. Також, одним з ризіків є можливість втрати ваших даних при глобальному збої сервісу.

Таблиця 1.1

Найвідоміші сервіси для хмарного зберігання файлів

№п/п	**Компанія**	**Назва продукту**	**Ємність безкоштновного плану**
1	Microsoft	OneDrive	5 GB безкоштовно
2	Google	Google Drive	15 GB безкоштовно
3	Dropbox	Dropbox	2 GB безкоштовно + можливість розширення за допомогою запрошень
4	Mega	Mega Cloud	50 GB безкоштовно

У результаті, ми бачимо є чимало сервісів від відомих техногігантів для даних цілей. У них є як безкоштовні тарифи, так і платні можливості для розширення кількості пам’яті, доступної для зберігання потрібної вам інформації та для доступу до неї з будь-якої точки світи за допомогою інтернету.

1.2 Принципи роботи хмарних сховищ

Хмарне сховище включає в себе як мінімум один сервер даних, до якого користувач підключається через Інтернет. Користувач відправляє файли вручну або в автоматичному режимі через Інтернет на сервер даних, який пересилає інформацію на кілька серверів. Збережені дані потім доступні через веб-інтерфейс.

У хмарних системах зберігання використовується величезна кількість серверів даних для забезпечення їх доступності. Таким чином, якщо один сервер вимагає обслуговування або виходить з ладу, користувач може бути впевнений, що дані були реплікуються в іншому місці для забезпечення доступності. Наприклад, в даний час Amazon AWS Cloud охоплює 55 зон доступності в 18 географічних регіонах.

У той час як дані в загальнодоступній хмарі реплікуються в різні фізичні місцеположення для забезпечення відмовостійкості і аварійного відновлення, основне або локальне розташування зазвичай ближче до об'єкту компанії, котрі використовують його, так що дані можуть оброблятися швидше і тоді з меншими витратами, скажімо, вибравши основне місце на півдорозі навколо земної кулі.

Тенденції управління хмарним сховищем продовжують розвиватися, і все більше компаній переходять на хмарні обчислення. Загальнодоступні хмари управляються постачальниками загальнодоступних хмарних послуг. Їх інфраструктура і послуги включають в себе:

- Сервери
- Місце зберігання
- Мережа
- Робота центрів обробки даних

Запропонуємо графічне представлення роботи хмарного сховища (рис. 1.1)

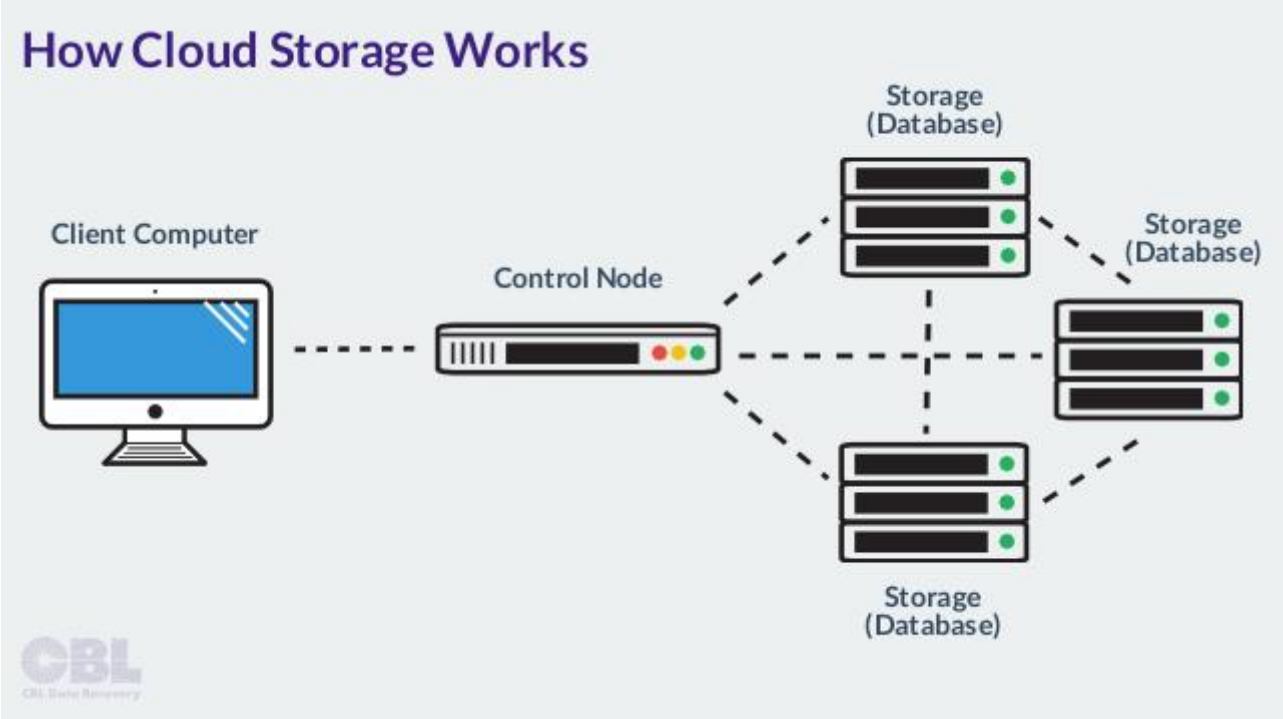

Рис.1.1 – Приблизна схема роботи хмарного схевища

Ресурси хмарного зберігання даних можуть бути надані наступними способами:

- Кінцевими користувачами, які використовують веб-інтерфейс, який оплачує пропускну здатність для кожної транзакції

- користувачами, які вказують заздалегідь визначену пропускну здатність, підготовлену заздалегідь для послуги. Потім клієнт платить за послугу щомісяця або платить фіксовану плату.

- Постачальником послуг, який розподіляє ресурси динамічно в міру необхідності. Оплата здійснюється на основі плати за використання.

Як зазначалося вище, ціни можуть варіюватися і можуть залежати від ряду факторів, у тому числі від постачальника послуг, необхідної ємності, необхідного часу і т.д.

1.3 Види хмарних сховищ

Існує чотири основних типи хмарного сховища:

1. Персональне хмарне сховище
2. Приватне хмарне сховище
3. Публічне хмарне сховище
4. Гібридне хмарне сховище.

Персональне хмарне сховище забезпечується підключеним до мережі пристроєм, який дозволяє користувачам зберігати різні типи особистих даних. Приклади хмарного сховища включають текст, графіку, фотографії, відео і музику. Користувач володіє пристроєм і контролює його, а також може отримати до нього доступ з будь-якого місця через Інтернет. Пристрій дійсно персональний хмарний накопичувач.

У *приватному хмарному сховищ*і використовуються локальні сервери зберігання, які знаходяться під контролем компанії, якій вони належать. Подібно загальнодоступному хмарного сховища і центрів обробки даних, приватна хмарне сховище використовує переваги віртуальних машин.

Приватні хмари, як правило, використовуються організаціями, яким потрібна гнучкість і масштабованість хмарного сховища, хоча і під прямим контролем і управлінням компанії, якій воно належить. Організації, стурбовані безпекою, можуть віддати перевагу управління своєю власною архітектурою хмарних систем зберігання даних, а не використанням публічного хмари.

Загальнодоступне хмарне сховище або *публічне хмарне сховище* доступно від стороннього постачальника в якості служби. Хмарне сховище Amazon AWS, хмарне сховище Microsoft Azure і хмарне сховище Google зазвичай користуються популярністю серед підприємств. Ці загальнодоступні хмарні опції доступні як сервіс. Інфраструктура створюється, належить, управляється і підтримується постачальниками хмарних сховищ. Багато веб-сайти хмарних сховищ також можна знайти в Інтернеті.

Гібридне хмарне сховище - це таке собі поєднання публічного хмари, приватного хмари і центру обробки даних, яке вважає за краще організація. Як правило, він об'єднує ресурси, якими володіє і управляє підприємство, з загальнодоступними хмарними службами зберігання, які управляються третьою стороною. Підприємства об'єднують два підходи, щоб збалансувати потребу в захисті критично важливих активів з перевагами гнучкості, масштабованості і вартості, які забезпечує загальнодоступне хмарне сховище.

Публічні та приватні хмари використовують переваги хмарних обчислень і технологій зберігання даних. Однак є деякі відмінності, які варто враховувати:

- Володіння і контроль - ресурси загальнодоступного хмарного сховища належать стороннім постачальникам послуг і контролюються ними; приватна хмарна інфраструктура належить і контролюється власними силами;

- Оновлення включені чи ні - загальнодоступне хмарне сховище включає в себе оновлення; у приватних хмар їх немає;

- Загальні і виділені ресурси - інфраструктура загальнодоступного хмарного сховища спільно використовується групою користувачів; приватна хмарне сховище призначене для компанії, яка володіє ним;

- Ступінь безпеки - хмарна безпека є спірним моментом. Деякі вважають публічні хмари менш безпечними, ніж приватні. Проте, організації з обмеженими IT-ресурсами можуть віддати перевагу покладатися на експертні знання постачальника послуг хмарного зберігання;

- Відмовостійкість - загальнодоступне хмарне сховище реплицирует дані; приватні хмари в одному місці можуть бути зруйновані в результаті стихійного лиха.

ВИСНОВОК ДО РОЗДІЛУ 1

У даному розділі був проведений невеличкий аналіз технології хмарного зберігання файлів.

Зокрема, проведено огляд сучасних хмарних сховищ від різних компаній. Також, проаналізовано спосіб роботи хмарних сховищ та надана схема роботи.

Сервіси хмарних сховищ надають багато переваг для зберігання ваших файлів та доступу до них з будь-якої точки світу. Але сторону безпеки ми вже розглянемо в наступних розділах.

РОЗДІЛ 2
ЗАСОБИ ДЛЯ СТВОРЕННЯ ДПХС

2.1 Інтернет речей (Internet of Things)

«Інтернет речей» (IoT) стає все більш зростаючої темою для розмов як на робочому місці, так і за його межами. Це концепція, яка може впливати не тільки на те, як ми живемо, але і на те, як ми працюємо. Але що таке «Інтернет речей» і який вплив він матиме на вас, якщо такий буде? Є багато складнощів навколо "Інтернету речей", але я хочу дотримуватися основ. Було проведено безліч технічних і політичних дискусій, але багато людей все ще намагаються зрозуміти, про що ці розмови.

Давайте почнемо з розуміння кількох речей. Широкосмуговий Інтернет стає все більш доступним, знижується вартість підключення, створюється все більше пристроїв з можливостями Wi-Fi і вбудованими в них датчиками, знижуються витрати на технології і стрімко зростає проникнення смартфонів. Всі ці речі створюють «ідеальний шторм» для IoT.

Так що ж таке Інтернет речей?Простіше кажучи, це концепція в основному підключення будь-якого пристрою з включенням і вимиканням до Інтернету (і / або один до одного). Це включає в себе все, від стільникових телефонів, кавоварок, пральних машин, навушників, ламп, переносних пристроїв і майже до всього, про що ви тільки можете подумати. Це також відноситься до компонентів машин, наприклад, реактивному двигуну літака або буровій вишці. Як я вже згадував, якщо він має перемикач включення і виключення, швидше за все, він може бути частиною IoT. Аналітична компанія Gartner говорить, що до 2020 року буде підключено більше 26 мільярдів пристроїв ... Це багато підключень (деякі навіть оцінюють це число набагато вище, більше 100 мільярдів). Інтернет речей являє собою гігантську мережу пов'язаних «речей» (в яку також входять люди).

Відносини будуть між людьми-людьми, людьми-речами і речами-речами.

Як це впливає на вас? Нове правило на майбутнє буде: «Все, що може бути пов'язано, буде пов'язано». Але з якого дива ви хочете, щоб так багато підключених пристроїв розмовляли один з одним? Є багато прикладів того, як це може виглядати або яка потенційна цінність. Скажімо, наприклад, що ви на шляху до зустрічі; Ваша машина може мати доступ до вашого календаря і вже знає, який маршрут вибрати. Якщо трафік інтенсивний, ваша машина може відправити повідомлення іншій особі, повідомивши його, що ви встигнете. Що якщо ваш будильник розбудить вас о 6 годині ранку, а потім повідомить вашу кавоварку, щоб почати варити для вас каву? Що, якщо ваше офісне обладнання знає, коли закінчується дефіцит витратних матеріалів, і автоматично замовляє більше? Що якщо переносна пристрій, який ви використовували на робочому місці, могло б сказати вам, коли і де ви були найбільш активні і продуктивні, і поділилося цією інформацією з іншими пристроями, які ви використовували під час роботи?

У більш широкому масштабі IoT можна застосовувати до таких речей, як транспортні мережі: «розумні міста», які можуть допомогти нам скоротити втрати і підвищити ефективність таких речей, як використання енергії; це допомагає нам зрозуміти і поліпшити те, як ми працюємо і живемо. Подивіться на зображення нижче (рис. 2.1), щоб побачити, як може виглядати щось подібне.

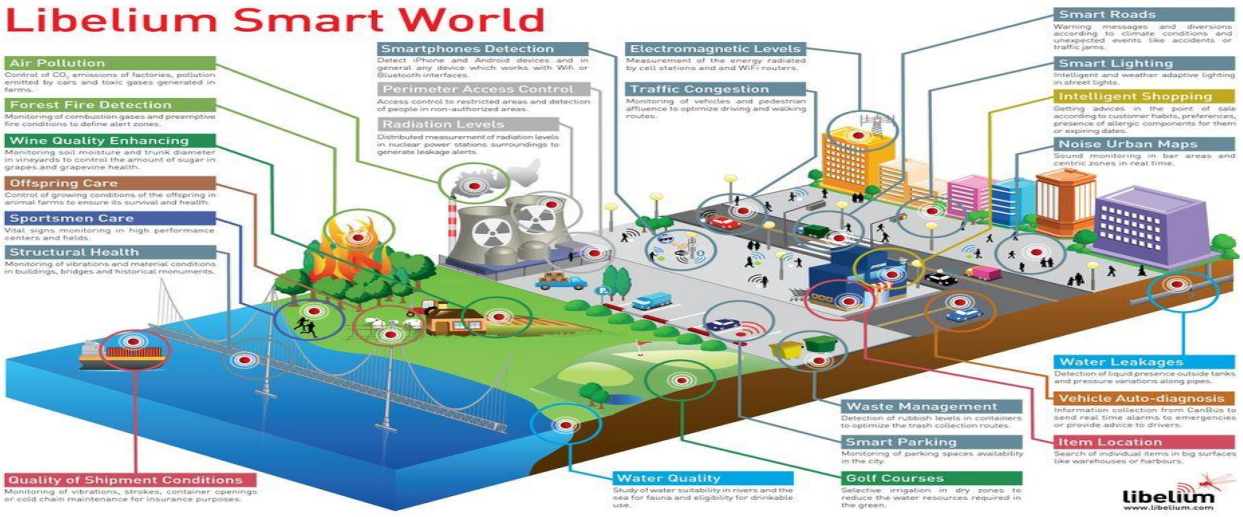

Рис. 2.1 – Приклад розумного міста Libelium

Реальність така, що IoT надає практично нескінченні можливості і зв'язку, про багатьох з яких ми навіть не можемо думати або повністю розуміти вплив сьогоднішнього дня. Неважко зрозуміти, як і чому IoT є такою гарячою темою сьогодні; це, безумовно, відкриває двері для багатьох можливостей, але і для багатьох проблем. Безпека - це велика проблема, яка часто піднімається. Коли мільярди пристроїв з'єднані разом, що можуть зробити люди, щоб забезпечити безпеку своєї інформації? Чи зможе хтось зламати тостер і отримати доступ до всієї вашої мережі? Інтернет речей також відкриває компаніям в усьому світі більше загроз безпеки. Тоді у нас є проблема конфіденційності та обміну даними. Це актуальна тема навіть сьогодні, тому можна тільки уявити, як розмови і проблеми будуть загострюватися, коли ми говоримо про багатомільярдний підключенні пристроїв. Інша проблема, з якою зіткнуться багато компаній, пов'язана з величезними обсягами даних, які будуть збирати всі ці пристрої. Компанії повинні знайти спосіб зберігання, відстеження, аналізу та аналізу величезної кількості даних, які будуть згенеровані.

І що тепер? Розмови про IoT (і протягом декількох років) відбуваються по всьому світу, коли ми прагнемо зрозуміти, як це вплине на наше життя. Ми також намагаємося зрозуміти, якими будуть численні можливості і проблеми, оскільки все більше пристроїв починають приєднуватися до IoT. На даний момент краще, що ми можемо зробити, - це пізнати себе про те, що таке IoT, і про те, яке потенційний вплив можна побачити на те, як ми працюємо і живемо.

2.2 Raspberry Pi

Raspberry Pi - це назва серії одноплатних комп'ютерів, створених фондом Raspberry Pi Foundation, британською добродійною організацією, мета якої - навчити людей інформатики і спростити доступ до інформатики.

Raspberry Pi був запущений в 2012 році, і з тих пір було випущено кілька ітерацій і варіацій. Оригінальний Pi мав одноядерний процесор 700 МГц і всього 256 МБ ОЗУ, а остання модельна лінійка має чотирьохядерний процесор 1,5 ГГц з 1/2/4 ГБ оперативної пам'яті. Основний цінової ціною для Raspberry Pi

завжди було 35 доларів, а для всіх моделей - 35 доларів або менше, включаючи Pi Zero, який коштує всього 5 доларів.

У всьому світі люди використовують Raspberry Pis для вивчення навичок програмування, створення апаратних проектів, автоматизації будинку і навіть використання їх в промислових додатках.

Raspberry Pi - це дуже дешевий комп'ютер з операційною системою Linux, але він також надає набір висновків GPIO (введення / виведення загального призначення), які дозволяють вам управляти електронними компонентами для фізичних обчислень і досліджувати Інтернет речей (IoT).

У Raspberry Pi було три покоління: Pi 1, Pi 2 і Pi 3, і, як правило, модель A і модель B більшості поколінь. Модель A є дешевшим варіантом і має тенденцію до зменшення ОЗУ і портів, таких як USB і Ethernet. Pi Zero є побічним продуктом оригінального покоління (Pi 1), зробленим ще менше і дешевше.

Перелік моделей:

- Pi 1 Model B (2012) - $35
- Pi 1 Model A (2013) - $25
- Pi 1 Model B+ (2014) - $35
- Pi 1 Model A+ (2014) - $20
- Pi 2 Model B (2015) - $35
- Pi Zero (2015) - $35
- Pi 3 Model B (2016) - $35
- Pi Zero W (2017) - $35
- Pi 3 Model B+ (2018) - $35
- Pi 3 Model A+ (2019) - $25

Нижче, на малюнку 2.2 ви можете побачити, як саме виглядають усі моделі Raspberry Pi:

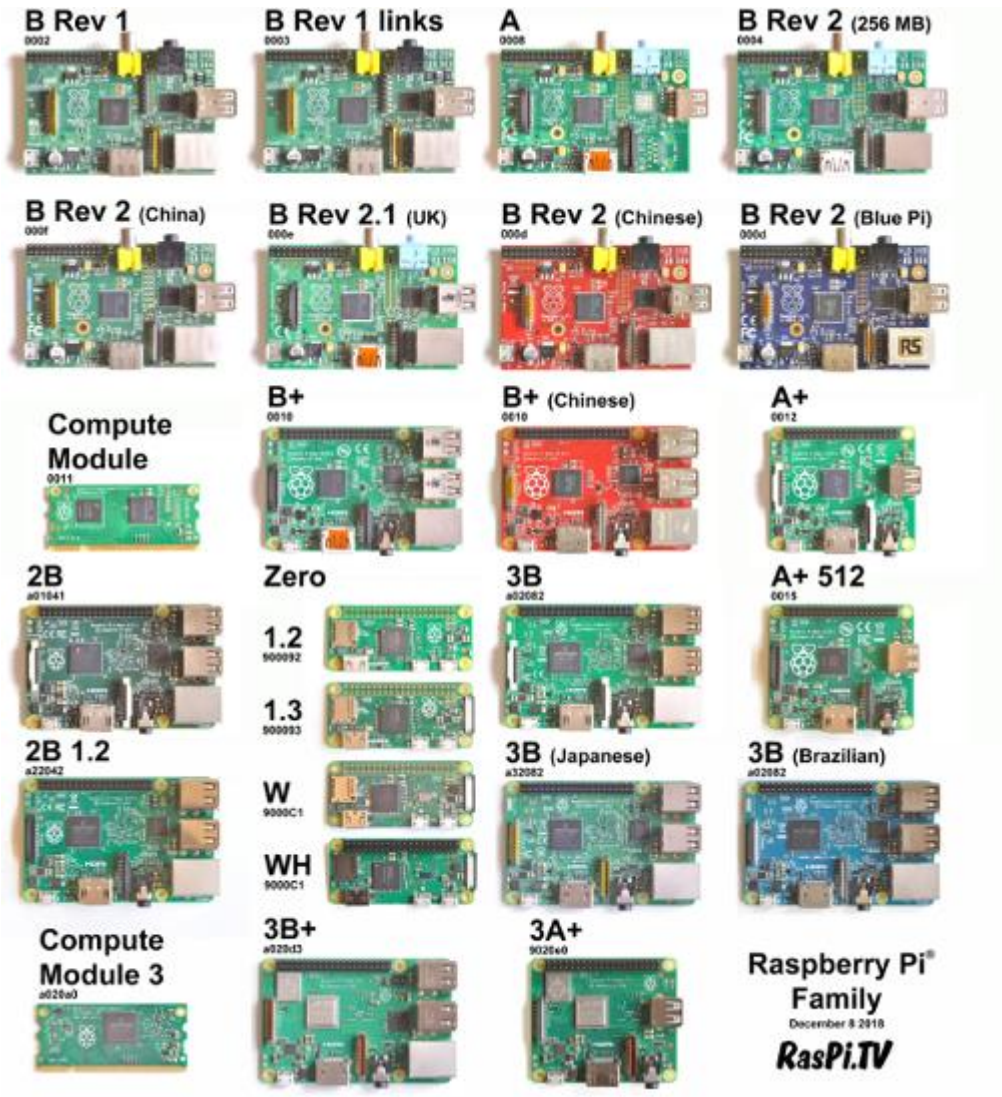

Рис. 2.2 – Модельний ряд Raspberry Pi

Фонд Raspberry Pi працює над тим, щоб передати можливості комп'ютерних і цифрових технологій людям усього світу. Це досягається шляхом надання недорогих, високопродуктивних комп'ютерів, які люди використовують для навчання, вирішення проблем і розваги. Він забезпечує охоплення і освіту, щоб допомогти більшій кількості людей отримати доступ до комп'ютерів і цифрових технологій - він розробляє безкоштовні ресурси, щоб допомогти людям дізнатися про комп'ютери та створювати речі за допомогою комп'ютерів, а також навчає викладачів, які можуть направляти інших людей до навчання.

Raspberry Pi працює в екосистемі з відкритим вихідним кодом: він працює під управлінням Linux (різні дистрибутиви), а його основна підтримувана операційна система, Raspbian, має відкритий вихідний код і використовує набір

програмного забезпечення з відкритим вихідним кодом. Фонд Raspberry Pi вносить свій внесок в ядро Linux і різні інші проекти з відкритим вихідним кодом, а також випускає велику частину свого власного програмного забезпечення в якості відкритого джерела.

Схеми Raspberry Pi випущені, але сама плата не є відкритим апаратним забезпеченням. Фонд Raspberry Pi покладається на дохід від продажу Raspberry Pis для своєї благодійної діяльності.

Що ви можете зробити з Raspberry Pi? Деякі люди купують Raspberry Pi, щоб навчитися програмуванню, і люди, які вже це вміють, використовують Pi, щоб навчитися програмувати електроніку для фізичних проектів. Raspberry Pi може відкрити вам можливості для створення власних проектів домашньої автоматизації, які популярні серед людей в співтоваристві відкритого вихідного коду, тому що вони дають вам контроль, а не використовують пропрієтарну закриту систему.

2.3 Операційна система Raspbian

Raspbian OS є рекомендованою операційною системою для Rasbperry Pi. Саме я буду використовувати модель Raspberry Pi 3B+. Вона є розігнаною версією 3B. Операційна система Raspbian поставляється з фірмовим інсталятором NOOBS (New Out Of Box Software). В даний час в NOOBS включені наступні операційні системи:

- Raspbian
- LibreELEC
- OSMC
- Recalbox
- Lakka
- RISC OS
- Screeny OSE
- Windows 10 IoT Core
- TLXOS

Починаючи з NOOBS v1.3.10 (вересень 2014 г.), в NOOBS за замовчуванням офлайн встановлюється тільки Raspbian. Інші можуть бути встановлені за допомогою під’єднання до мережі Інтернет прямо в процесі встановлення.

NOOBS доступний в двох форматах: змішаний (офлайн інсталятор та онлайн інстялятор) або тільки онлайн інсталятор. Повна версія включає ОС Raspbian, тому її можна встановити з SD-карти в автономному режимі, тоді як для використання NOOBS Lite або встановлення будь-якої іншої операційної системи потрібне підключення до Інтернету.

Зверніть увагу, що образ операційної системи в повній версії може бути застарілим, якщо випущена нова версія ОС, але при підключенні до Інтернету вам буде показана можливість завантаження останньої версії, якщо буде доступна його новіша.

2.3.1 Підготовка до встановлення операційної системи

Для того, щоб встановити дану операційну систему, не потрібно бути досвіченим експертом. Усе, що від вас потребується, це підключення до інтернету, комп’ютер на операційній системі Windows, MacOS або Linux.

Першим етапом є завантаження інсталятора операційної системи – NOOBS. Він може бути завантежений у вигляді .zip архіву з офіційного сайту https://www.raspberrypi.org/. Поки йде завантаження, вам потрібно підгтувати SD-картку, на яку буде і записана операційна система.

Все, що зберігається на SD-карті, буде перезаписано під час форматування. Так що якщо на SD-карті, на яку ви хочете встановити Raspbian, є будь-які файли, наприклад, в старішої версії Raspbian ви можете спочатку зберегти ці файли, щоб не втратити їх назавжди.

1. Завантажте SD Formatter для Windows або Mac.
2. Дотримуйтесь інструкцій для установки програмного забезпечення.
3. Вставте SD-карту в роз'єм для комп'ютера або ноутбука.
4. У SD Formatter виберіть свою SD-карту і відформатуйте карту.

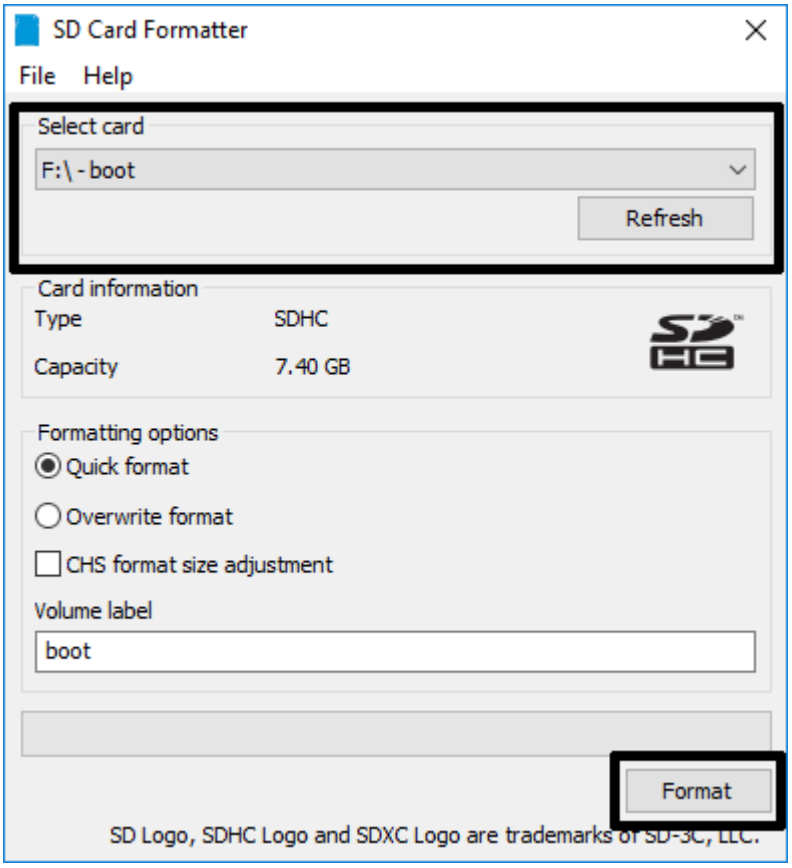

Рис.2.3 – вікно програми SD Formatter

Потім вам потрібно буде витягти файли з zip-архіву NOOBS, який ви завантажили з веб-сайту Raspberry Pi. Знайдіть завантажений архів - за замовчуванням він повинен знаходитися в папці «Завантаження». Двічі клацніть по ньому, щоб витягти файли, і залиште відкрите вікно Explorer / Finder відкритим.

Тепер відкрийте інше вікно Explorer / Finder і перейдіть до SD-карті. Найкраще розташувати два вікна поруч. Виберіть всі файли в папці NOOBS і перетягніть їх у вікно SD-карти, щоб скопіювати їх на карту.

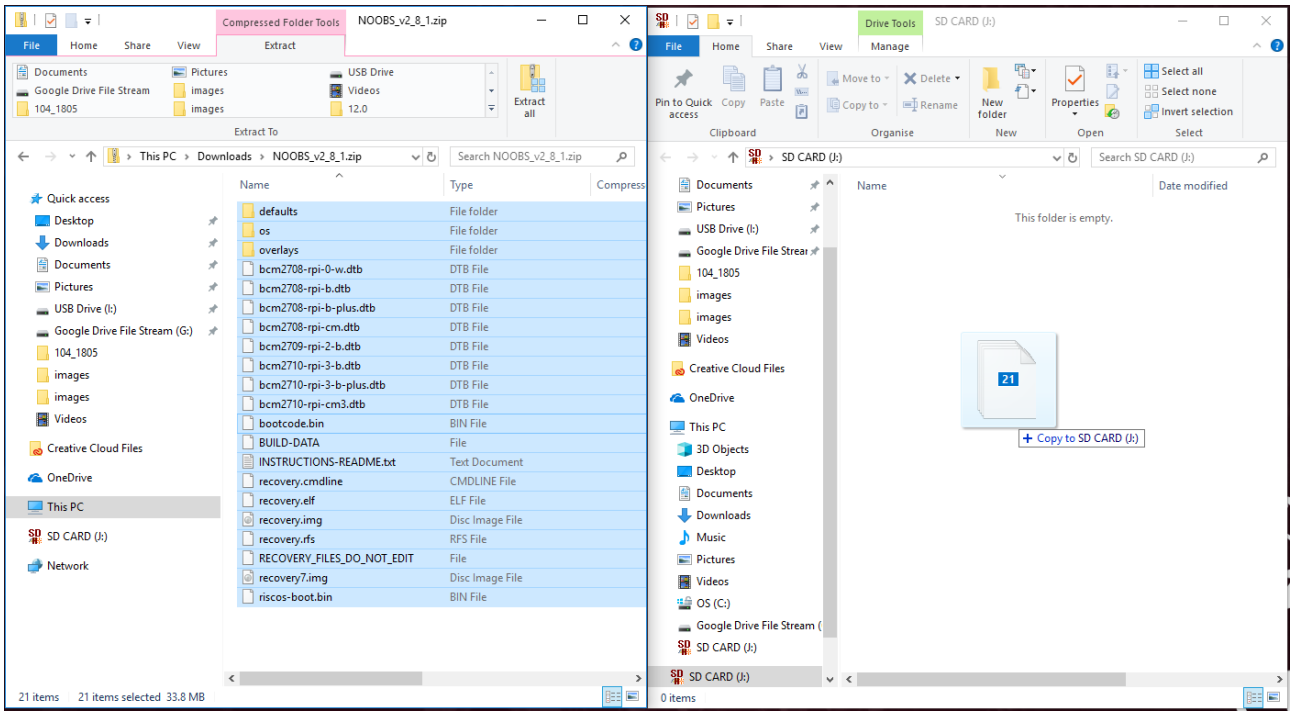

Рис.2.4 – Процес копіювання файлів операційної системи

Після того, як всі файли були скопійовані, ви можете витягти SD-карту.

Тепер підключіть все до вашого Raspberry Pi. Важливо зробити це в правильному порядку, щоб всі ваші компоненти були в безпеці.

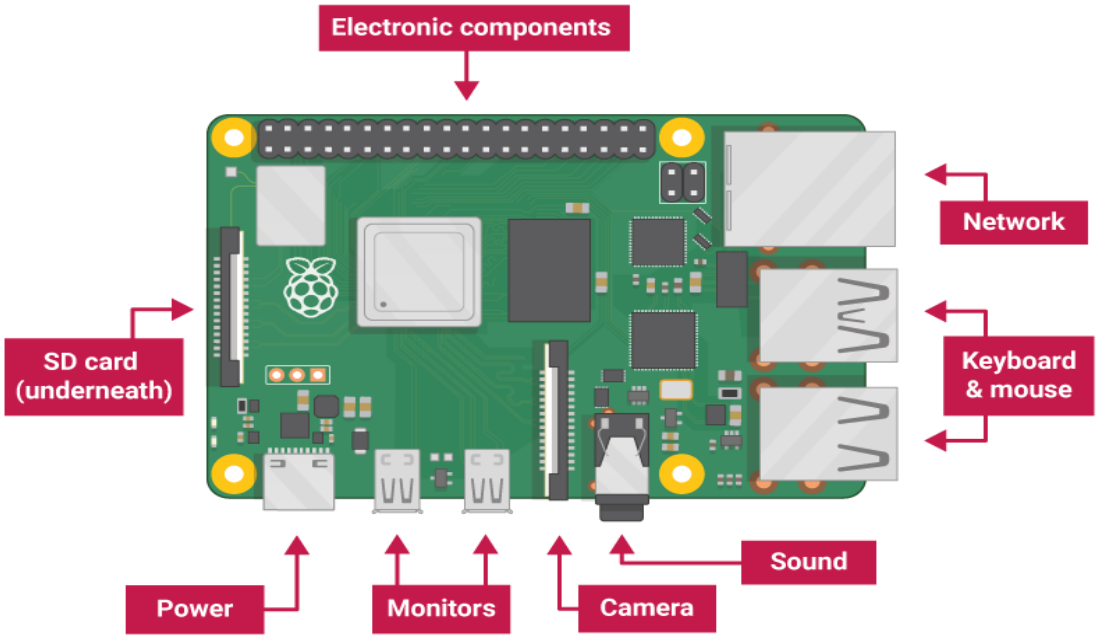

Рис.2.5 – Схема підключення пристроїв до плати

Вставте SD-карту, яку ви налаштували з допомогою Raspbian (через NOOBS), в слот для карти microSD на нижньому боці Raspberry Pi. Після цього, увімкніть живлення та зачекайте включення пристрою.

2.3.2 Встановлення та перший запуск

Коли ви вперше запускаєте Raspberry Pi, з'явиться віконце Welcome to Raspberry Pi, яке проведе вас через початкове налаштування.

Рис.2.6 – Віконце Welcome to Raspberry Pi

Натисніть Далі, щоб почати налаштування. Встановіть країну, мову і часовий пояс, потім знову натисніть «Далі». Вам буде запропоновано встановити пароль, далі – під’єднатися до мережі інтернет за допомогою WiFi (в разі, якщо не буде прямого з’єднання через Ethernet), вибір операційної системи (в нашому випадку це Raspbian Lite). Rasbpian Lite є такою ж самої ОС, як і звичайна Raspbian, але не має графічного середовища. Тільки термінал, більше нам і не потрібно. Пройшовши усі пункти, ви перейдете до процесу встановлення, який після завершення запропонує перезавантажити пристрій.

Після перезавантаження пристрою, ми успішно потрапляємо в операційну систему. В нашом випадку це термінал. Логінемось стандартними login – pi, password – raspberry. Потім ви можете змінити ці дані. Одразу йдемо у налаштування системи за допомогою команди *“sudo raspi-config”*. Після цього ми потрапляємо у меню конфігурації системи, яке має наступний вигляд:

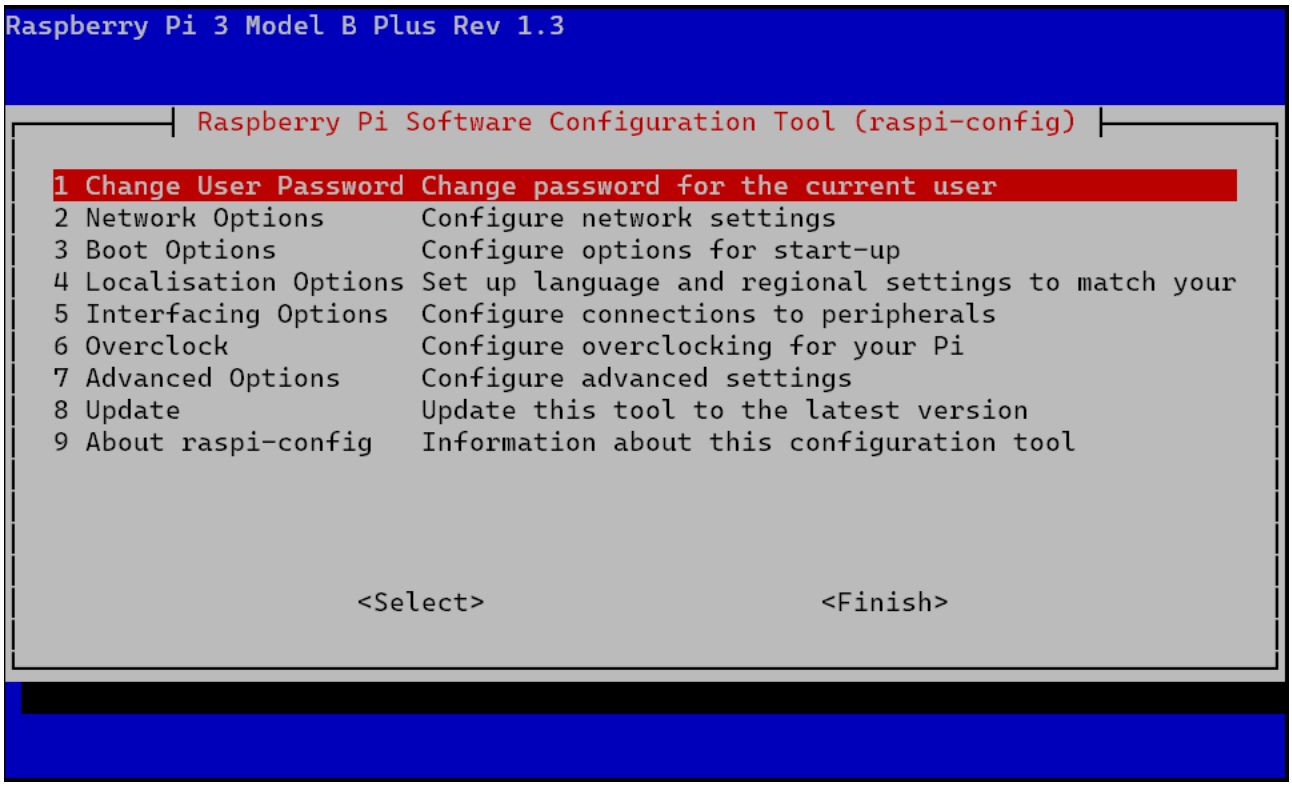

Рис.2.7 – Віконце *raspi-config*

Переходимо у пункт Interfacing Options, потім у пункт SSH. Вмикаємо дану функцію та перезавантажуємо наш міні-ПК. Тепер ми можемо віддалено підключатися до нашого Raspberry Pi і нам більше не потрібен монітор, клавіатура. Вимикаємо усю перферію і залишаємо тільки живлення.

2.4 SSH технологія

SSH, також відомий як Secure Shell або Secure Socket Shell, є мережевим протоколом, який надає користувачам, особливо системним адміністраторам, безпечний доступ до комп'ютера по незахищеній мережі. SSH також відноситься до набору утиліт, які реалізують протокол SSH. Secure Shell забезпечує сувору аутентифікацію і зашифровану передачу даних між двома комп'ютерами, підключеними по відкритій мережі, такий як Інтернет. SSH широко використовується мережевими адміністраторами для віддаленого управління системами і додатками, що дозволяє їм входити в мережу з іншого комп'ютера, виконувати команди і переміщати файли з одного комп'ютера на інший.

SSH відноситься як до криптографічним мережевого протоколу, так і до набору утиліт, які реалізують цей протокол. SSH використовує модель клієнт-сервер, поєднуючи клієнтську програму захищеної оболонки, кінець, в якому відображається сеанс, з сервером SSH, кінець, в якому виконується сеанс. Реалізації SSH часто включають підтримку прикладних протоколів, використовуваних для емуляції терміналу або передачі файлів. SSH також можна використовувати для створення безпечних тунелів для інших протоколів додатків, наприклад, для безпечного запуску графічних сеансів X Window System. Сервер SSH за замовчуванням прослуховує стандартний порт 22 протоколу управління передачею даних (TCP).

Хоча в якості облікових даних можна використовувати SSH зі звичайним ідентифікатором користувача і паролем, SSH частіше використовує пари відкритих ключів для аутентифікації хостів один з одним. Окремі користувачі як і раніше повинні використовувати свій ідентифікатор користувача та пароль (або інші методи аутентифікації) для підключення до самого віддаленого хосту, але локальний комп'ютер і віддалений комп'ютер автентифіковані окремо один від одного. Це досягається шляхом генерації унікальної пари відкритих ключів для кожного хоста в повідомленні; Для одного сеансу потрібні дві пари відкритих ключів: одна пара відкритих ключів для аутентифікації віддаленого комп'ютера на локальному комп'ютері і друга пара відкритих ключів для аутентифікації локального комп'ютера на віддаленому комп'ютері.

З'єднання SSH використовувалися для захисту безлічі різних типів зв'язку між локальним комп'ютером і віддаленим хостом, включаючи безпечний віддалений доступ до ресурсів, віддалене виконання команд, доставку виправлень і оновлень програмного забезпечення та інші завдання адміністрування або управління.

Функції, які включає SSH, включають:

- Безпечний віддалений доступ до мережевих систем або пристроїв з підтримкою SSH для користувачів, а також для автоматизованих процесів;

- безпечні і інтерактивні сеанси передачі файлів;

- автоматична і безпечна передача файлів;

- безпечний введення команд на віддалених пристроях або системах;

- безпечне управління компонентами мережевої інфраструктури.

SSH може використовуватися в інтерактивному режимі для включення термінальних сеансів і повинен використовуватися замість менш захищеною програми Telnet. SSH також зазвичай використовується в сценаріях і другом програмному забезпеченні, щоб дозволити програмам і системам віддалено і безпечно отримувати доступ до даних і інших ресурсів.

2.4.1 Підключення до Raspberry Pi за допомогою SSH

Для підключення до нашої Rapsberry Pi ми будемо використовувати віддалений доступ за допомогою SSH протоколу. Але, для початку нам потрібен SSH клієнт для Windows. На щастя, ми можемо встановити PuTTY. Завантажуємо це ПЗ з офіційного сайту www.putty.org та встановлюємо. Після встановлення та запуску клієнту нам потрібно потрібно ввести локальну IP адресу нашого Raspberry Pi (в моєму випадку це 192.168.31.99) та вибрати тип з'єднання SSH. Якщо ж ви не знаєте IP адресу свого Raspberry Pi, то ви можете його дізнатися за допомогою команди *hostname -I* на своєму пристрої Raspberry Pi.

Після введення усіх данних в клієнті PuTTY натисніть Connect та ви отримаєте віддалений доступ до пристрою. Спочатку вас попросять ввести логін та пароль пристрою, до якого ви підключились:

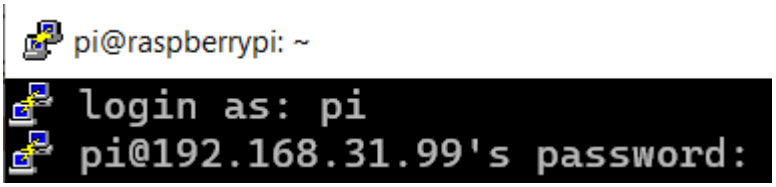

Рис.2.8 – Віконце логіну через SSH у клінті PuTTY

Увівши усі дані, ви побачите повідомлення про успішне підключення та матимете змогу керувати пристроєм:

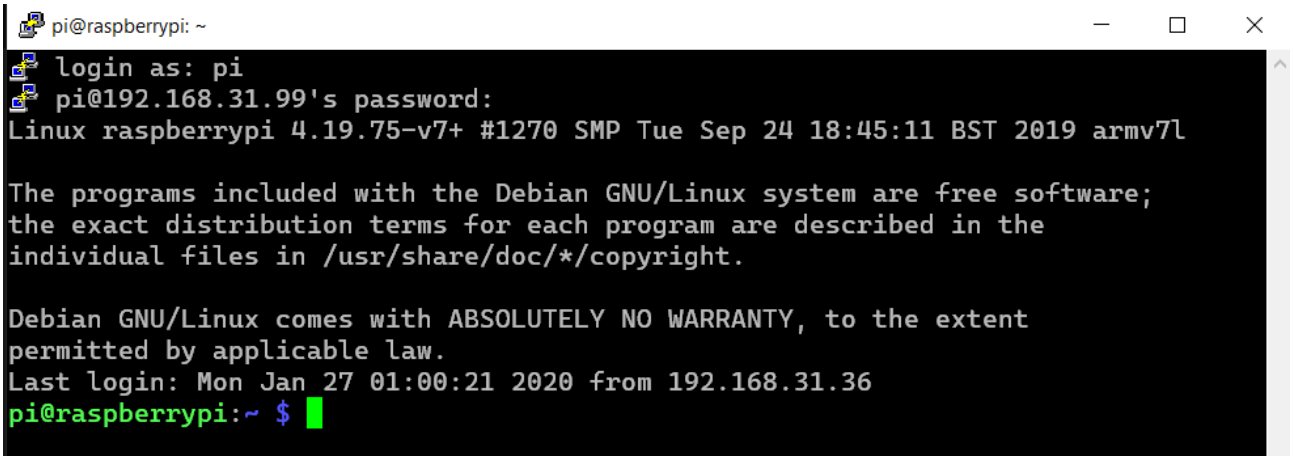

Рис.2.9 – Вдале підключення через SSH

Тепер ви готові до роботи з пристроєм!

2.5 nginx

Для створення хмарного сховища нам просто необхідно мати встановленний веб-сервер на нашому пристрої. Ми будемо використовувати nginx.

Nginx, вимовляється як «engine-ex», є веб-сервером з відкритим вихідним кодом, який з моменту свого первісного успіху в якості веб-сервера тепер також використовується в якості зворотного проксі-сервера, HTTP-кеша і балансувальника навантаження.

Деякі відомі компанії, що використовують Nginx, включають Autodesk, Atlassian, Intuit, T-Mobile, GitLab, DuckDuckGo, Microsoft, IBM, Google, Adobe, Salesforce, VMWare, Xerox, LinkedIn, Cisco, Facebook, Target, Citrix Systems, Twitter, Apple , Intel і багато інших.

Спочатку Nginx був створений Ігорем Сисоєва, перший публічний випуск якого відбувся в жовтні 2004 року. Ігор спочатку задумував програмне забезпечення як відповідь на проблему C10k, яка пов'язана з проблемою продуктивності при обробці 10 000 одночасних підключень.

Оскільки його коріння лежить в масштабній оптимізації продуктивності,

Nginx часто перевершує інші популярні веб-сервери в тестах продуктивності, особливо в ситуаціях зі статичним контентом і / або високими паралельними запитами, тому Kinsta використовує Nginx для забезпечення свого хостингу.

2.5.1 Як працює Nginx?

Nginx створений, щоб запропонувати низьке використання пам'яті і високий паралелізм. Замість того, щоб створювати нові процеси для кожного веб-запиту, Nginx використовує асинхронний, керований подіями підхід, коли запити обробляються в одному потоці. З Nginx один головний процес може керувати кількома робочими процесами. Майстер підтримує робочі процеси, в той час як робітники виконують фактичне опрацювання. Оскільки Nginx є асинхронним, кожен запит може виконуватися працівником одночасно, не блокуючи інші запити.

Деякі загальні риси, які спостерігаються в Nginx:

- Зворотній проксі з кешуванням
- IPv6
- Балансування навантаження
- підтримка FastCGI з кешуванням
- WebSockets
- Обробка статичних файлів, індексних файлів і автоматична індексація
- TLS / SSL з SNI
-

2.6 SSL сертифікат

Для безпечного з’єднання з нашою хмарою, нам також необхідно буде мати сертифікат безпеки – SSL.

SSL або Secure Sockets Layer - це протокол безпеки Інтернету, заснований на шифруванні. Вперше він був розроблений Netscape в 1995 році з метою забезпечення конфіденційності, аутентифікації і цілісності даних в інтернет-комунікаціях. SSL є попередником сучасного шифрування TLS, використовуваного сьогодні.

Як працює SSL?

- Щоб забезпечити високу ступінь конфіденційності, SSL шифрує дані, які передаються через Інтернет. Це означає, що будь-який, хто спробує перехопити ці дані, побачить тільки спотворене поєднання символів, яке майже неможливо розшифрувати.
- SSL ініціює процес аутентифікації, званий рукостисканням, між двома пристроями, щоб переконатися, що обидва пристрої дійсно є тими, ким себе називають.
- SSL також використовує цифровий підпис даних для забезпечення цілісності даних, перевіряючи, що дані не були підроблені до того, як вони досягнуть свого передбачуваного одержувача.

2.6.1 OpenSSL

Щоб зрозуміти OpenSSL, вам також необхідно зрозуміти дві його широкі цілі. По-перше, він служить інструментарієм для протоколів Secure Sockets Layer (SSL) і Transport Layer Security (TLS). По-друге, це бібліотека криптографії загального призначення для додатків, що забезпечують зв'язок, що відбувається по комп'ютерних мережах. Це важливо для розкриття особистості сторін, які здійснюють зв'язок між двома точками в Інтернеті, і захисту систем від прослуховування. OpenSSL спочатку написаний на C, але має оболонки (програми або дані, які створюють інші програми або дані, щоб вони могли працювати без збоїв), що підтримують безліч інших мов. Він широко використовується на веб-серверах, причому більше 60% веб-серверів використовують їх в 2017 році.

Бібліотека OpenSSL містить інструменти, необхідні для виконання таких завдань:

- Генерація закритих ключів для RSA (Rivest-Shamir-Adleman, асиметричні алгоритми шифрування)
- Створення запиту на підпис сертифіката (CSR)
- Виконання шифрування / дешифрування, а також управління сертифікатами

Отже, OpenSSL можна описувати повний спектр як платформу, яка надає безліч функцій утиліт, а також реалізує основні криптографічні функції. Це робить його важливим елементом інтернет-безпеки і криптографії.

2.7 PHP

PHP позначає Hypertext Preprocessor (немає, абревіатура не слід за ім'ям). Це серверний мова сценаріїв з відкритим вихідним кодом, який використовується для розробки веб-додатків. Під мовою сценаріїв ми розуміємо програму, засновану на сценаріях (рядки коду), написану для автоматизації завдань.

Що означає відкритий вихідний код? Подумайте про виробника автомобілів, який робить секрет своїх моделей дизайну і технологічних інновацій доступним для всіх, хто зацікавлений. Ці деталі дизайну і технології можуть бути перерозподілені, змінені і прийняті без страху будь-яких правових наслідків. Світ сьогодні, можливо, розробив дивовижний суперкар!

Веб-сторінки можуть бути розроблені з використанням HTML. В HTML виконання коду виконується в браузері користувача (на стороні клієнта). З іншого боку, при використанні мови сценаріїв на стороні PHP він виконується на сервері до того, як він потрапляє в веб-браузер користувача.

PHP може бути вбудований в HTML, і він добре підходить для веб-розробки і створення динамічних веб-сторінок для веб-додатків, додатків електронної комерції і додатків баз даних. Він вважається дружнім мовою, здатним легко з'єднуватися з MySQL, Oracle та іншими базами даних.

2.7.1 Використання PHP

Сценарії PHP можуть використовуватися в більшості відомих операційних систем, таких як Linux, Unix, Solaris, Microsoft Windows, MAC OS і багатьох інших. Він також підтримує більшість веб-серверів, включаючи Apache і IIS. Використання PHP надає веб-розробникам свободу вибору своєї операційної системи і веб-сервера.

У PHP сценарії на стороні сервера є основною областю діяльності. Скрипти на стороні сервера з PHP включають в себе:

- PHP Parser: програма, яка перетворює вихідний і читається людиною код в формат, більш зручний для розуміння комп'ютером.
- Веб-сервер: програма, яка виконує файли, які формують веб-сторінки з призначених для користувача запитів.
- Веб-браузер: додаток, що використовується для відображення контенту в World Wide Web

В цьому випадку, використовуючи тільки синтаксичний аналізатор PHP, сценарій PHP може виконуватися без серверної програми або браузера. Таке використання сценарію PHP зазвичай використовується для простих завдань обробки тексту, таких як планувальники завдань.

PHP також може бути використаний для створення клієнтських додатків, таких як настільні додатки. Настільні додатки зазвичай характеризуються графічним інтерфейсом користувача. Володіючи знаннями в використанні розширених функцій PHP, таких як PHP-GTK, ці клієнтські програми можуть бути розроблені.

2.7.2 Переваги PHP

PHP це програмне забезпечення з відкритим вихідним кодом. Це означає, що він знаходиться у вільному доступі для поширення на внесення змін, на відміну від інших мов програм. Існує також команда розробників PHP, готових надати будь-яку необхідну технічну підтримку. Їх внесок також означає, що мова постійно вдосконалюється і оновлює свої основні функції.

PHP простий у використанні і зміні, і він дуже сумісний з провідними операційними системами і веб-браузерами. Це значно спрощує розгортання додатків в різних операційних системах і браузерах. Як згадувалося раніше, підтримуються такі платформи, як Linux, Windows, Solaris і MAC OS. Це також відноситься до веб-серверів, таким як Apache, IIS і інші.

Оскільки такі технології, як HTML, Get і Post, вбудовані, PHP дуже добре працює в веб-розробці і простий у використанні.

З його характеристиками відкритого вихідного коду приходить світ різноманітності і велика кількість бібліотек і розширень. З відкритим вихідним кодом з'являється можливість для модифікації коду, і завдяки цьому були розроблені і зроблені вільно доступними численні призначені для користувача розширення і компоненти.

Відкритий вихідний код дійсно має значення. Спільнота розробників і технічна підтримка в сфері PHP полегшують пошук технічних рішень. Як то кажуть, кожен питання ставилося раніше, і у кого-то є рішення. Спільнота робить це можливим. Якщо вам потрібен конкретний скрипт, існує висока ймовірність того, що інший користувач PHP може розробити щось подібне, якщо не ідентичне.

Широке використання PHP з відкритим вихідним кодом означає, що набагато більше людей знайомі з мовою, що значно полегшує пошук технічних знань.

2.8 Linux

Основним нашим інструментом в побудові хмарного сховища буде термінал операційної системи Raspbian, яка є Linux дистрибутивом. Тому, я пропоную зробити огляд самого Linux.

Операційна система Linux всюди - від смартфонів до автомобілів, суперкомп'ютерів і побутової техніки, від домашніх комп'ютерів до корпоративних серверів.

Linux існує з середини 1990-х років і з тих пір досяг призначеної для користувача бази, яка охоплює всю земну кулю. Linux насправді скрізь: він є в ваших телефонах, ваших термостатах, в ваших автомобілях, холодильниках, пристроях Roku і телевізорах. Він також керує здебільшого Інтернету, всіх 500 кращих суперкомп'ютерів світу і світових фондових бірж.

Але крім того, що Linux є платформою для настільних комп'ютерів, серверів і вбудованих систем по всьому світу, Linux є однією з найнадійніших,

безпечних і доступних операційних систем.

Як і Windows, iOS і Mac OS, Linux є операційною системою. Фактично, одна з найпопулярніших платформ на планеті Android працює на операційній системі Linux. Операційна система - це програмне забезпечення, яке керує всіма апаратними ресурсами, пов'язаними з вашим настільним комп'ютером або ноутбуком. Простіше кажучи, операційна система управляє зв'язком між вашим програмним забезпеченням і обладнанням. Без операційної системи (ОС) програмне забезпечення не буде працювати.

Операційна система Linux складається з декількох частин:

1. Завантажувач - програмне забезпечення, яке управляє процесом завантаження вашого комп'ютера. Для більшості користувачів це буде просто заставка, яка спливає і в кінцевому підсумку йде для завантаження операційної системи.
2. Ядро - це одна частина цілого, яка насправді називається «Linux». Ядро є ядром системи і управляє процесором, пам'яттю і периферійними пристроями. Ядро - найнижчий рівень ОС.
3. Система ініціалізації - це підсистема, яка завантажує простір користувача і відповідає за управління демонами. Systemd - одна з найбільш широко використовуваних систем ініціалізації. що також виявляється одним з найбільш суперечливих. Саме система init управляє процесом завантаження після передачі початкового завантаження з завантажувача (тобто GRUB або GRand Unified Bootloader).
4. Daemons. Це фонові служби (друк, звук, планування і т. Д.), які запускаються або під час завантаження, або після входу на робочий стіл.
5. Графічний сервер - це підсистема, яка відображає графіку на вашому моніторі. Зазвичай його називають X-сервером або просто X.
6. Середовище робочого столу - це та частина, з якою користувачі взаємодіють. Існує безліч середовищ робочого столу на вибір

(GNOME, Cinnamon, Mate, Pantheon, Enlightenment, KDE, Xfce і т. Д.). Кожна середу робочого столу включає в себе вбудовані додатки (такі як файлові менеджери, інструменти настройки, веб-браузери і ігри).

7. Додатки - середовища робочого столу не пропонують повний набір додатків. Як і Windows і MacOS, Linux пропонує тисячі і тисячі високоякісних програмних продуктів, які легко знайти і встановити. Більшість сучасних дистрибутивів Linux (докладніше про це нижче) включають інструменти, подібні App Store, які централизуют і спрощують установку додатків. Наприклад, в Ubuntu Linux є Ubuntu Software Center (ребрендинг програмного забезпечення GNOME? Малюнок 1), який дозволяє вам швидко виконувати пошук серед тисяч додатків і встановлювати їх з одного централізованого місця розташування.

2.8.1 Навіщо використовувати Linux?

Це єдине питання, яке задають більшість людей. Навіщо вивчати абсолютно іншу обчислювальну середу, коли операційна система, яка поставляється з більшістю настільних комп'ютерів, ноутбуків і серверів, працює нормально?

Щоб відповісти на це питання, я б поставив інше питання. Ця операційна система, яку ви зараз використовуєте, дійсно працює? Або ви стикаєтеся з перешкодами, такими як віруси, шкідливі програми, уповільнення, збої, дорогий ремонт і ліцензійні збори?

Якщо ви боретеся з вищевикладеним, Linux може бути ідеальною платформою для вас. Linux перетворився в одну з найбільш надійних комп'ютерних екосистем на планеті. Об'єднайте цю надійність з нульовою вартістю входу, і ви отримаєте ідеальне рішення для настільної платформи.

Це вірно, нульова вартість входу ... як безкоштовно. Ви можете встановити Linux на будь-яку кількість комп'ютерів, не сплачуючи ні цента за ліцензування програмного забезпечення або сервера.

Давайте поглянемо на вартість сервера Linux в порівнянні з Windows

Server 2016. Ціна стандартної версії Windows Server 2016 становить 882,00 дол. США (купується безпосередньо у Microsoft). Це не включає клієнтську ліцензію (CAL) і ліцензії на програмне забезпечення, яке вам може знадобитися (наприклад, база даних, веб-сервер, поштовий сервер і т. Д.). Наприклад, ліцензія на одного користувача для Windows Server 2016 коштує 38 доларів США. Наприклад, якщо вам потрібно додати 10 користувачів, це буде на 388,00 доларів більше за ліцензування серверного програмного забезпечення. З сервером Linux все це безкоштовно і легко встановити. Насправді, для установки повноцінного веб-сервера (який включає в себе сервер бази даних) достатньо всього лише кількох кліків миші або команд (подивіться «Easy LAMP Server Installation», щоб зрозуміти, наскільки це просто).

Якщо нульовою вартістю недостатньо, щоб перемогти вас, то як щодо операційної системи, яка буде працювати без проблем, поки ви використовуєте її? Я використовую Linux вже майже 4 роки (як у якості настільної, так і серверної платформи), і у мене не було проблем з вимагачами, шкідливими програмами або вірусами. Linux зазвичай набагато менше вразливий для таких атак. Що стосується перезавантаження сервера, вони необхідні тільки при оновленні ядра. Незвично, що сервер Linux працює роками без перезавантаження. Якщо ви прямуєте регулярним рекомендованим оновлень, стабільність і надійність практично гарантовані.

2.8.2 Linux Terminal

Консоль Linux - це системна консоль, вбудована в ядро Linux (системна консоль - це пристрій, який отримує всі повідомлення і попередження ядра і дозволяє вхід в систему в режимі одного). Консоль Linux дозволяє ядру і іншим процесам відправляти текстовий висновок користувачеві і отримувати текстове введення від користувача. Користувач зазвичай вводить текст з клавіатури комп'ютера і читає текст, що виводиться на монітор комп'ютера. Ядро Linux підтримує віртуальні консолі - консолі, які логічно розділені, але мають доступ до однієї і тієї ж фізичної клавіатурі і дисплею. Консоль Linux (і віртуальні консолі Linux) реалізовані підсистемою VT ядра Linux і не залежать від будь-

якого програмного забезпечення призначеного для користувача простору. Це відрізняється від емулятора терміналу, який являє собою процес користувацького простору, який емулює термінал, і зазвичай використовується в середовищі графічного відображення.

Консоль Linux була однією з перших функцій ядра і була спочатку написана Лінус Торвальдс в 1991 році (див. Історію Linux). Існує дві основні реалізації: кадровий буфер і текстовий режим. Реалізація кадрового буфера за замовчуванням використовується в сучасних дистрибутивах Linux і разом з налаштуванням режиму ядра забезпечує підтримку на рівні ядра для апаратного забезпечення дисплея і такі функції, як відображення графіки під час завантаження системи. Застаріла реалізація текстового режиму використовувалася в ПК-сумісних системах з графічними картами CGA, EGA, MDA і VGA. Hi-x86 архітектури використовували режим кадрового буфера, тому що їх відеокарти не реалізували текстовий режим. Консоль Linux використовує растрові шрифти фіксованого розміру, моноширинних шрифти, зазвичай за умовчанням 8x16 пікселів на символ.

Консоль Linux є додатковою функцією ядра, і більшість вбудованих систем Linux не включають її. Ці системи зазвичай надають альтернативний користувальницький інтерфейс (наприклад, веб-інтерфейс) або відразу завантажуються в графічний користувальницький інтерфейс і використовують його в якості основного засобу взаємодії з користувачем. Інші реалізації консолі Linux включають консоль Брайля для підтримки оновлюваних дисплеїв Брайля і консоль послідовного порту.

ВИСНОВОК ДО РОЗДІЛУ 2

У другому розділи ми дізналися дуже багато про пристрій Rapberry Pi. Цей міні-комп’ютер, який коштує всього $35 та має розмір кредитної картки, надає вам неймовірні можливості: від створення пристроїв Інтернету Речей до налагодження власного веб серверу. І ці можливості тільки зростають.

Також, ми дізналися з вами про ядро Linux, термінал та трошки їхньої історії.

Також, дуже необхідна технологія для роботи віддалено - SSH була доволі детально описана в цьому розділі та були показані її можливості, а також процес налаштування.

Серверна технологія nginx також зайняла вагому частину розділу та є необхідним компонентом мого проєкту.

Ну, і на останок, були розглянутий необхідний компонент – PHP версії 7.2. Ми навчилися писати команди в терміналі, оновлювати пакети, встановлювати їх, переходити між теками та налаштовувати конфіг файли. .

РОЗДІЛ 3
ВСТАНОВЛЕННЯ НЕОБХІДНИХ КОМПОНЕНТІВ. ВИКОРИСТАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ OWNCLOUD.

3.1 Підготовка до встановлення усіх необхідних компонентів

Для початку, нам необхідно оновити нашу операційну систему та усі системні пакети. Для цього, ми під'єднуємося до нашої Raspberry Pi за допомогою SSH клієнту.

Логінимося та робимо оновлення списку ситемних пакетів:

```
sudo apt-get update
```

Оновлюємо список пакетів і встановлюємо Oracle Java 8.

```
sudo apt-get upgrade
```

Погоджуємося з оновленням усіх пакетів та чекаемо завершення оновлення.

Після вдалого оновлення, прописуємо команду:

```
sudo raspi-config
```

В конфігуряціях нам потрібно пересвідчитися в тому, що у internationalization options -> change local налаштуваннях вказано Locale -> en_US.UTF8.

В цьому ж конфігураторі, у меню Advanced options -> Memory split змінюємо виділену під відео пам'ять до 16.

Тепер ми готові до встановлення основних компонентів для нашого хмарного сховища

Додайте користувача www-data в групу www-data:

```
sudo usermod -a -G www-data www-data
```

3.1.1 Встановлення компонентів

Наступна команда допоможе нам встановити усі необхідні компоненти для роботи хмари, такі як nginx, openSSL, php 7.2:

```
sudo apt-get install nginx openssl ssl-cert php7.2-cli php7.2-sqlite3 php7.2-gd php7.2-common php7.2-cgi sqlite3 php-pear curl libapr1 libtool curl libcurl4-openssl-dev php7.2-xml php7.2 php7.2-dev php7.2-curl php7.2-gd php7.2-fpm memcached php-memcache php7.2-zip php7.2-intl php7.2-mbstring varnish
```

Тепер нам потрібно створити сертифікат SSL, ви можете зробити це, виконавши наступну команду:

```
    sudo openssl req $@ -new -x509 -days 730 -nodes -out /etc/nginx/cert.pem -keyout /etc/nginx/cert.key
```

На додаток до SSL-сертифікату нам також необхідно створити власний файл dhparam. Цей файл допомагає забезпечити безпеку наших з'єднань SSL, за замовчуванням він буде використовувати з'єднання за замовчуванням, яке буде не таким безпечнішим.

Щоб створити DH-параметр довжиною 4096 байт, запустіть наступну команду на Raspberry Pi. Цей процес займе досить багато часу, до 16 годин. Додавання прапора -dsaparam до команди допоможе прискорити процес, але, можливо, менш безпечно:

```
    sudo openssl dhparam -out /etc/nginx/dh4096.pem 4096
```

Тепер нам потрібно налаштувати chmod на три сертифікати, які ми тільки що згенерували.

```
    sudo chmod 600 /etc/nginx/cert.pem

    sudo chmod 600 /etc/nginx/cert.key

    sudo chmod 600 /etc/nginx/dh4096.pem
```

Давайте очистимо файл конфігурації сервера, так як ми будемо копіювати і вставляти в нього нашу власну версію.

```
    sudo sh -c "echo '' > /etc/nginx/sites-available/default"
```

3.1.2 Конфігурації серверних компонентів

Тепер давайте налаштуємо конфігурацію веб-сервера так, щоб він правильно запускав Owncloud.

```
sudo nano /etc/nginx/sites-available/default
```

Я буду використовувати наступні налаштування для серверу:

```
upstream php-handler {
    server unix:/var/run/php/php7.2-fpm.sock;
 }

 server {
    listen 80;
    server_name _;

    #Allow letsencrypt through
    location /.well-known/acme-challenge/ {
        root /var/www/owncloud;
    }

    # enforce https
    location / {
        return 301 https://$host$request_uri;
    }
 }

 server {
    listen 443 ssl http2;
    server_name _;

    ssl_certificate /etc/nginx/cert.pem;
```

```
        ssl_certificate_key /etc/nginx/cert.key;

        ssl_session_timeout 5m;
        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        ssl_ciphers                                    'ECDHE-RSA-AES128-GCM-
SHA256:AES256+EECDH:AES256+EDH';
        ssl_dhparam /etc/nginx/dh4096.pem;
        ssl_prefer_server_ciphers on;
        keepalive_timeout    70;
        ssl_stapling on;
        ssl_stapling_verify on;

        add_header X-Content-Type-Options nosniff;
        add_header X-Frame-Options "SAMEORIGIN";
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Robots-Tag none;
        add_header X-Download-Options noopen;
        add_header X-Permitted-Cross-Domain-Policies none;

        root /var/www/owncloud/;

        location = /robots.txt {
            allow all;
            log_not_found off;
            access_log off;
        }

        # The following 2 rules are only needed for the user_webfinger app.
        # Uncomment it if you're planning to use this app.
        #rewrite ^/.well-known/host-meta /public.php?service=host-meta last;
        #rewrite  ^/.well-known/host-meta.json  /public.php?service=host-meta-json
```

```
last;

        location = /.well-known/carddav {
           return 301 $scheme://$host/remote.php/dav;
        }
        location = /.well-known/caldav {
           return 301 $scheme://$host/remote.php/dav;
        }

        # set max upload size
        client_max_body_size 512M;
        fastcgi_buffers 8 4K;
        fastcgi_ignore_headers X-Accel-Buffering;

        gzip off;

        error_page 403 /core/templates/403.php;
        error_page 404 /core/templates/404.php;

        location / {
           rewrite ^ /index.php$uri;
        }

        location ~ ^/(?:build|tests|config|lib|3rdparty|templates|data)/ {
           return 404;
        }
        location ~ ^/(?:\.|autotest|occ|issue|indie|db_|console) {
           return 404;
        }

        location                                                    ~
```

```
^/(?:index|remote|public|cron|core/ajax/update|status|ocs/v[12]|updater/.+|ocs-provider/.+|core/templates/40[34])\.php(?:$|/) {
        fastcgi_split_path_info ^(.+\.php)(/.*)$;
        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        fastcgi_param SCRIPT_NAME $fastcgi_script_name;
        fastcgi_param PATH_INFO $fastcgi_path_info;
        fastcgi_param HTTPS on;
        fastcgi_param modHeadersAvailable true;
        fastcgi_param front_controller_active true;
        fastcgi_read_timeout 180;
        fastcgi_pass php-handler;
        fastcgi_intercept_errors on;
        fastcgi_request_buffering off; #Available since NGINX 1.7.11
    }

    location ~ ^/(?:updater|ocs-provider)(?:$|/) {
        try_files $uri $uri/ =404;
        index index.php;
    }

    location ~ \.(?:css|js)$ {
        try_files $uri /index.php$uri$is_args$args;
        add_header Cache-Control "max-age=15778463";
        # Before enabling Strict-Transport-Security headers please read into this topic first.
        #add_header Strict-Transport-Security "max-age=15552000; includeSubDomains";
        add_header X-Content-Type-Options nosniff;
        add_header X-Frame-Options "SAMEORIGIN";
```

```
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Robots-Tag none;
        add_header X-Download-Options noopen;
        add_header X-Permitted-Cross-Domain-Policies none;
        access_log off;
    }

    location ~ \.(?:svg|gif|png|html|ttf|woff|ico|jpg|jpeg|map)$ {
        add_header Cache-Control "public, max-age=7200";
        try_files $uri /index.php$uri$is_args$args;
        access_log off;
    }
}
```

Після цього, зберігаємо файл конфігурації та виходимо з редактору файлу.

3.1.3 Налаштування PHP

На цьому етапі нам треба прописати конфігурації у PHP. Для цього, переходимо до редагування файлу з налаштуваннями завдяки наступній команді:

```
sudo nano /etc/php/7.2/fpm/php.ini
```

У цьому файлі робимо пошук та шукаємо:

1. upload_max_filesize
2. post_max_size

Дані параметри задають розміри для одного файлу на завантаження та зберігання. Для кожного з параметрів виставляємо потрібне значення. В моєму випадке, це значення буде = 2024M. Зберігаємо зміни та виходимо з режиму редагування файлу.

3.1.4 Налаштування файлу підкачки

SWAP або файл підкачки - це простір на диску, яке використовується, коли обсяг фізичної оперативної пам'яті заповнений. Коли в системі Linux закінчується ОЗУ, неактивні сторінки переміщуються з ОЗУ в простір підкачки.

Простір підкачки може приймати форму виділеного розділу підкачки або файлу підкачки. У більшості випадків при запуску Linux на віртуальній машині розділ підкачки відсутній, тому наш єдиний варіант - створити файл підкачки.

Використовуємо наступну команду:

```
sudo nano /etc/dphys-swapfile
```

В даному файлі шукаємо строку CONF_SWAPSIZE та присвоюємо їй значення = 512.

Після цього, нам потрібно перезавантажити нашу Raspberry Pi за допомгою команди:

```
sudo reboot
```

3.1.5 Базові конфігурації сервісу OwnCloud

Як тільки Pi увімкнеться знову, нам потрібно буде встановити Owncloud на Raspberry Pi. Створюємо папку:

```
sudo mkdir -p /var/www/owncloud
```

Переходимо у папку www:

```
cd /var/www/
```

Завантажуємо останню версію OwnCloud:

```
curl   https://download.owncloud.org/community/owncloud-10.3.2.tar.bz2   | sudo tar -jxv
```

Задаємо права власника для теки:

```
sudo chown -R www-data:www-data /var/www
```

Перед більш детальною конфігурацією пропоную вам ознайомитися з самим продуктом OwnCloud, щоб мати уявлення про нього.

3.1.6 Огляд OwnCloud

ownCloud - це відкрита платформа для підвищення продуктивності і безпеки в цифровому співпраці .ownCloud - це захищений корпоративний файлообмінник. Такий же простий у використанні, як споживчий продукт, але розміщений в вашому дата-центрі. ownCloud пропонує неперевершену прозорість, безпеку і контроль і може гнучко інтегруватися в існуючу середу. У той же час користувачі можуть швидко і легко отримувати доступ до файлів компанії з будь-якого місця і з будь-якого пристрою. Це підвищує безпеку і продуктивність.

ownCloud - це сервер з відкритим вихідним кодом для синхронізації і обміну файлами з відкритим вихідним кодом. Як і «великі хлопчики» Dropbox, Google Drive, Box та інші, ownCloud дозволяє отримувати доступ до файлів, календаря, контактів і іншими даними. Ви можете синхронізувати всі (або частина цього) між вашими пристроями та обмінюватися файлами з іншими. Але ownCloud може зробити набагато більше, ніж його пропрієтарні, розміщені на чужому комп'ютері конкуренти.

Масштабований кластер ownCloud Pi. Оскільки ownCloud є відкритим вихідним кодом, ви можете вибирати між самостійним розміщенням на своєму власному сервері або орендою місця у постачальника, якому довіряєте, - немає необхідності розміщувати свої файли у великій компанії, яка зберігає їх і знає де. Знайдіть тут деяких провайдерів ownCloud або скачайте пакети або віртуальну машину для свого власного сервера тут.

Найкреативніші речі, які ми бачили, це кластер Banana Pi і кластер Raspberry Pi. Незважаючи на те, що масштабованість ownCloud часто використовується для розгортання сотень тисяч користувачів, деякі люди використовують його в іншому напрямку, об'єднуючи кілька крихітних систем для створення надшвидкого ownCloud. Престижність!

Тримайте ваші паролі синхронізованими. Щоб спростити розширення ownCloud, ми зробили його надзвичайно модульним і створили магазин

додатків ownCloud. Там ви можете знайти такі речі, як музичні та відеоплеєри, календарі, контакти, додатки для підвищення продуктивності, ігри, додаток для замальовок і багато іншого.

Важко вибрати тільки один додаток з майже 200 доступних, але управління паролями, безумовно, є унікальною функцією. Існує не менше трьох додатків, що надають цю функцію: паролі, безпечний контейнер і Passman.

Зберігайте свої файли там, де ви хочете. Зовнішнє сховище дозволяє підключити існуюче сховище даних до ownCloud, дозволяючи отримувати доступ до файлів, що зберігаються на FTP, WebDAV, Amazon S3 і навіть Dropbox і Google Drive через один інтерфейс.

«Великим хлопчикам» подобається створювати свої власні маленькі обгороджені сади - користувач Box може співпрацювати тільки з іншими користувачами Box; і якщо ви хочете поділитися своїми файлами з Google Диска, ваш партнер потребує облікового запису Google, або вони не можуть багато зробити. Із зовнішнім сховищем ownCloud ви можете подолати ці бар'єри.

Дуже креативне рішення - додати Google Drive і Dropbox в якості зовнішнього сховища. Ви можете безперешкодно працювати з файлами і обмінюватися ними з іншими за допомогою простого посилання - для роботи з вами не потрібно обліковий запис!

Завантажуйте файли. Оскільки ownCloud є відкритим вихідним кодом, люди надають цікаві функції, не обмежуючись корпоративних вимог. Наші учасники завжди дбали про безпеку і конфіденційності, тому ownCloud ввів такі функції, як захист суспільної посилання паролем і установка дати закінчення роками раніше, ніж будь-хто інший.

Сьогодні ownCloud має можливість налаштовувати загальну посилання як доступну для читання і запису, що означає, що відвідувачі можуть легко редагувати файли, якими ви з ними ділитеся (захищені паролем чи ні), або завантажувати нові файли на свій сервер без необхідності реєструватися в

інший веб сервіс, який хоче отримати свої особисті дані.

Це дуже зручно, коли люди хочуть поділитися з вами великим файлом. Замість того, щоб завантажувати його на сторонній сайт, відправляти вам посилання і змушувати вас йти туди і завантажувати його (часто вимагає входу в систему), вони можуть просто завантажити його в надану вами загальну папку, і ви можете перейти на працювати відразу

Отримайте безкоштовне безпечне сховище. Ми вже говорили про те, скільки наших учасників дбають про безпеку та конфіденційність. Ось чому ownCloud має додаток, яке може шифрувати і розшифровувати збережені дані.

Використання ownCloud для зберігання ваших файлів в Dropbox або на Google Диску руйнує саму ідею відновлення контролю над вашими даними і збереження їх у таєм ниці. Додаток шифрування змінює це. Зашіфровивая дані перед відправкою цим провайдерам і розшифровуючи їх при добуванні, ваші дані в безпеці, як кошенята.

3.1.7 Фінальні налаштування серверної частини OwnCloud

Після того, як ми розібралися, що ж таке OwnCloud та встановили дане ПЗ на наш пристрій, нам потрібно оновити конфігураційні файли самого OwnCloud. Тепер нам потрібно відкрити файл *.user.ini*, щоб застосувати деякі зміни, які ми внесли до цього часу.

```
sudo nano /var/www/owncloud/.user.ini
```

В цьому файлі редагуємо строки *upload_max_filesize*, *post_max_size* та *memory_limit*. Вони мусять мати наступний вигляд:

```
upload_max_filesize=2000M

post_max_size=2000M

memory_limit=2000M
```

Після цього ми зможемо підключитися до Owncloud по IP-адресою вашого PI.

Перед налаштуванням облікового запису адміністратора мені потребується підключити зовнішній диск, щоб було достатньо місця на диску для мого сервера Raspberry Pi Owncloud.

3.1.8 Під'єднаня зовнішнього HDD для зберігання файлів

Налаштування зовнішнього диска повинна бути відносно простою, але іноді все працює не так коректно, як слід було б. Але, у мене все вийшло одразу.

По-перше, якщо у вас є диск NTFS, нам потрібно встановити пакет NTFS, ввівши наступне:

```
sudo apt-get install ntfs-3g
```

Тепер нам потрібно створити теку, яку ми зможемо під'єднати:

```
sudo mkdir /media/owncloudddrive
```

Тепер нам потрібно отримати GID, UID і UUID, так як вони нам знадобляться найближчим часом. Введіть наступну команду для GID:

```
id -g www-data
```

Так само робимо і для UID:

```
id -u www-data
```

Також, якщо ми отримаємо UUID жорсткого диска, Pi запам'ятає цей диск, навіть якщо ви підключите його до іншого USB-порту:

```
ls -l /dev/disk/by-uuid
```

Тут нам потрібно скопіювати собі ID пристрою, який вказано синім кольором:

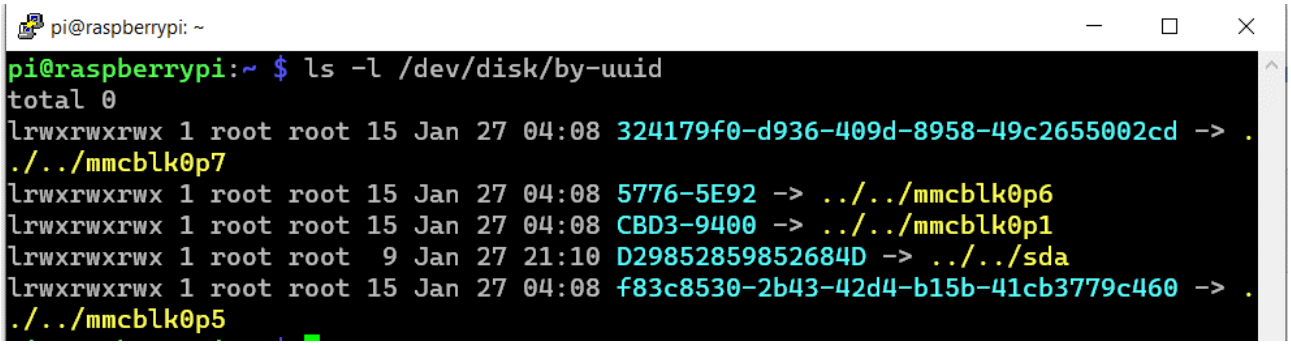

Рис.3.1 – Відображення списку пристроїв

В моєму випадку це D29852859852684D.

Тепер давайте додамо наш диск в файл fstab, щоб він завантажився з правильними дозволами. Переходимо в редагування файлу:

```
sudo nano /etc/fstab
```

Тепер ми додаємо наступний рядок в кінець файлу, оновлюючи UID, GUID і UUID значеннями, які ми отримали вище. (Наступне повинно бути в одному рядку):

```
UUID=DC72-0315 /media/owncloud drive auto nofail,uid=33,gid=33,umask=0027,dmask=0027,noatime 0 0
```

Перезавантажуємо Raspberry Pi, і диски повинні бути автоматично змонтовані. Якщо вони встановлені, у нас все добре.

3.2 Welcome екран та перший вхід до OwnCloud

Нарешті ми налаштували нашу Raspberry Pi та можемо перейти до інтерфейсу самої хмари. На данний момент вона доступна в нас лише в локальній мережі.

Відкриваємо нову вкладку в браузері та вписуємо локальну IP адресу моєї Raspberry Pi. Можемо загадати її за допомогою команди hostname -I в терміналі. В моєму випадку це 192.168.31.99.

Після переходу ми побачимо вітальне вікно зі старотовими налаштування для нас. Нам потрібно буде задати ім’я адміністратора(користувача) хмарного сховища, пароль для доступу, а також папку з даними. Також, є додаткове поле з можливістю вибору бази даних для хмарного сховища. Але, для мене воно не активне, та за замовченням у мене буде використовуватися SQLite. Причиною тому є те, що я не встановлював іншу базу даних. Базової SQLite мені достатньо.

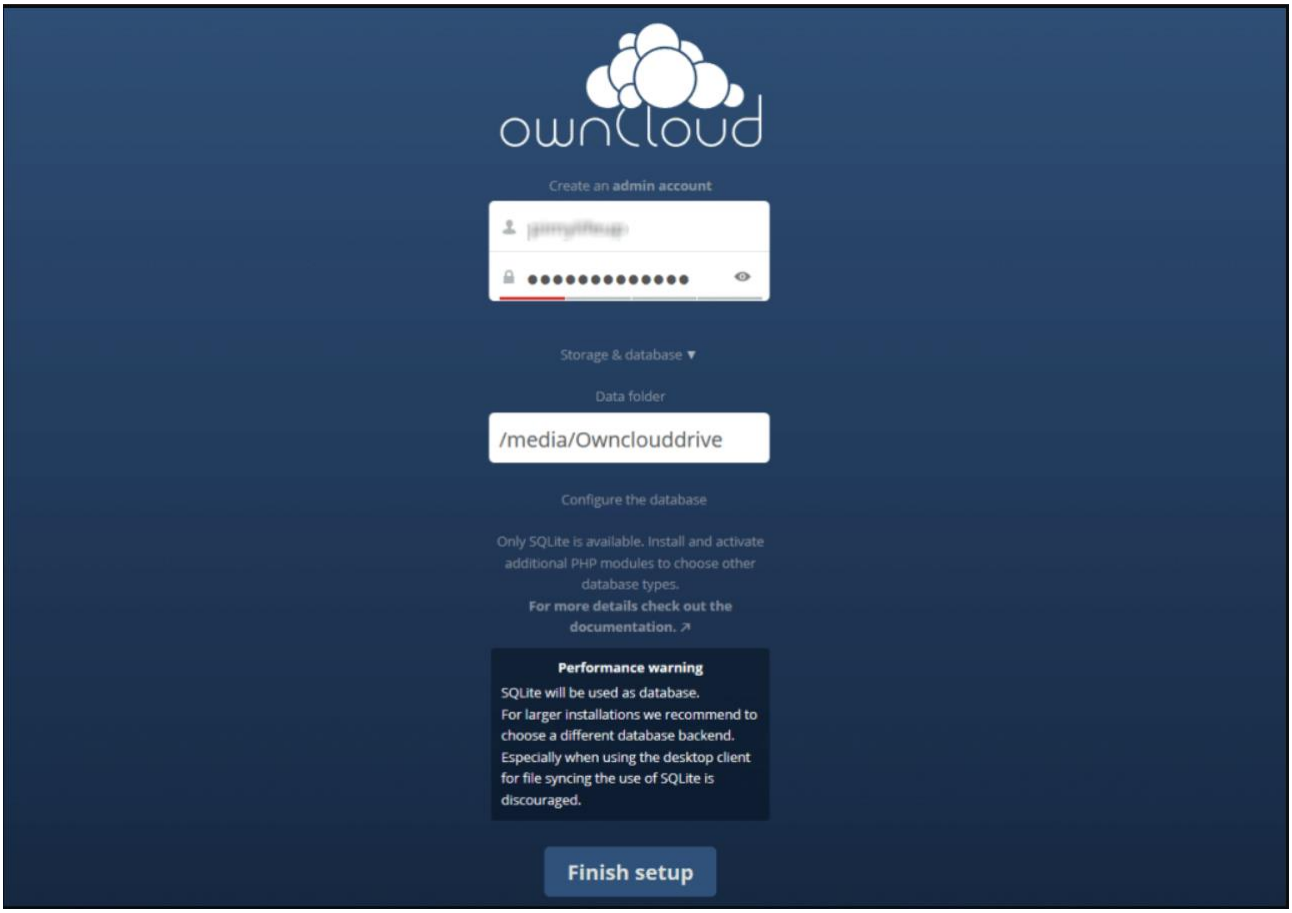

Рис.3.2 – Стартове вікно OwnCloud

Після введення усіх потрібних даних вам потрібно натиснути кнопку “Finish Setup”, яка дасть змогу завершити налаштування з використанням даних, які ви ввели.

Після цього ви будете направлені на вікно входу, де вам потрібно буде ввести логін та пароль. Готово! Ви можете виристовувати власну хмару в локальній мережі. Також, хочу повідомити, что OwnCloud надає безкоштновні додатки для мобільних пристроїв на базі iOS та Android, а таккож безкоштновні клієнти для синхронізація для ПК на операційних система Windows, MacOS та Linux. Ви можете завантажити їх на офіційному сайті проекту ownCloud.

3.3 Мережеве налаштування для доступу за межами локальної мережі

3.3.1 Перенаправлення портів

Для того, що б ми мали змогу під’єднуватися до нашої хмари за межами локальної мережі, тобто, з будь-якої точки світу, нам потрібно налштувати перенаправлення портів.

Для цього нам потрібно буде змінити деякі настройки на роутері. На комп'ютері, який підключений до локальної мережі, встановіть з'єднання з сторінці адміністратора маршрутизатора через веб-браузер.

1. IP-адреса маршрутизатора зазвичай дорівнює 192.168.1.1 або 192.168.1.254. У моєму випадку це адреса 192.168.31.1.
2. Введіть ім'я користувача і пароль для маршрутизатора. За замовчуванням це зазвичай admin & admin.
3. На сторінці адміністрування роутера перейдіть на forwarding-> virtual server
4. На цій сторінці введіть наступне
   - Service Port: 80
   - IP-адреса: 192.168.31.99
   - Internal Port: 80
   - Протокол: ALL
   - Статус: включений
5. Ці настройки будуть маршрутизувати трафік, призначений для порту, зазначеного на порт Raspberry Pi.
6. Тепер ви зможете підключитися до додатка на Raspberry Pi за межами вашої мережі.

Кращий спосіб перевірити правильність переадресації порту - це змусити одного підключитися або вийти куди-небудь за межі локальної мережі (VPN може цього домогтися).

3.3.2 Перенаправлення портів на Raspberry Pi

Якщо ваш інтернет-провайдер надає вам динамічний IP-адресу (часто змінюється IP-адреса), то, ймовірно, варто налаштувати динамічний DNS Raspberry Pi (іноді в них є функція маршрутизаторів). Цей метод означає, що ви завжди зможете підключитися до додатка на Пі, навіть якщо ваш зовнішній IP-адреса зміниться. У цьому випадку потрібно встановити DDclient, і це досить простий процес, який не займе багато часу для установки.

Для початку нам потрібно оновити всі пакети в базі операційної системи:

```
sudo apt-get update
```

Після цього можемо встановлювати DDclient та компоненти:

```
sudo apt-get install ddclient libjson-any-perl
```

Після всього цього нам тепер потрібно буде замінити цю версію DDclient більш нової. Ця новіша версія забезпечить кращу підтримку таких служб, як Cloudflare або NOIP. Для завантаження останньої версії введіть наступні команди:

```
wget https://files.pimylifeup.com/portforwarding/ddclient-3.9.0.tar.gz

tar -zxvf ddclient-3.9.0.tar.gz
```

Після того, як завантаження і витяг закінчені, ми будемо використовувати більш новий двійковий файл, скопіювавши його поверх поточного за допомогою наступної команди:

```
sudo cp -f ddclient-3.9.0/ddclient /usr/sbin/ddclient
```

Через зміни, внесені в найостаннішу версію DDclient, розташування файлу конфігурації змінилося. Ми будемо використовувати наступні команди, щоб змінити становище тієї, яка була при першій установці:

```
sudo mkdir /etc/ddclient

sudo mv /etc/ddclient.conf /etc/ddclient
```

Як тільки ddclient завершить установку, ми підемо і відредагуємо файл конфігурації, щоб внести кілька важливих змін. Використовуйте команду нижче, щоб запустити редактор для файлу:

```
sudo nano /etc/ddclient/ddclient.conf
```

Нам потрібно буде додати кілька рядків в цей файл, і ми будемо використовувати ці ж рядки в кожній конфігурації. Вони визначають, чи використовувати SSL і як отримати зовнішній IP-адреса Raspberry Pi.

Додайте наступне нижче коментаря заголовка в файлі, а також видаліть все інше в файлі. Ми додамо більше до файлу пізніше з документації No-IP або Cloudflare.

```
use=web, web=checkip.dyndns.com/, web-skip='IP Address'

ssl=yes
```

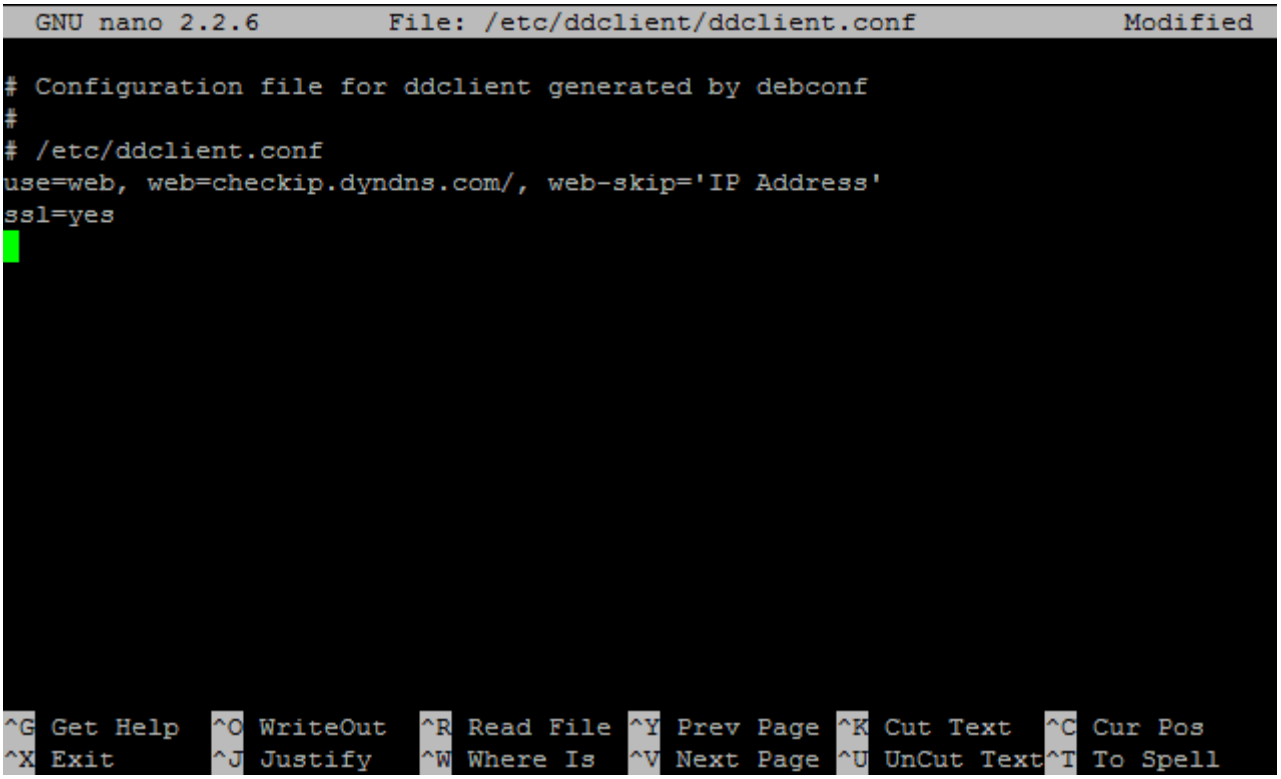

Рис.3.3 – вигляд конфігурації

Добре, тепер, коли це зроблено, ми можемо перейти до наступних кроків по налаштуванню ddclient з провайдером динамічного DNS.

3.4 DNS

Система доменних імен (DNS) є однією з основ інтернету, проте більшість людей поза мережею, ймовірно, не усвідомлюють, що використовують її кожен день, щоб виконувати свою роботу, перевіряти електронну пошту або витрачати час на свої смартфони.

За своєю суттю DNS - це каталог імен, які збігаються з числами. Числа, в даному випадку це IP-адреси, які комп'ютери використовують для зв'язку один з одним. У більшості описів DNS використовується аналогія з телефонною книгою, яка підійде людям старше 30 років, які знають, що таке телефонна книга.

Якщо вам менше 30 років, уявіть, що DNS - це список контактів вашого смартфона, який зіставляє імена людей з їх номерами телефонів і адресами електронної пошти. Потім помножте цей список контактів на всіх інших на планеті.

Коли Інтернет був дуже і дуже маленьким, людям було легше порівнювати конкретні IP-адреси з певними комп'ютерами, але це тривало недовго, коли до зростаючої мережі приєдналося більше пристроїв і людей. Крім створення каталогу для всіх цих пристроїв, були використані слова, що дозволяють людям підключатися до різних сайтах; для більшості людей запам'ятати слова легше, ніж запам'ятати конкретні набори чисел. Для доступу до веб-сайту все ще можна ввести певний IP-адреса в браузер.

Як працюють DNS-сервери

Каталог DNS, який відповідає імені та номерами, що не знаходиться в одному місці в якомусь темному куточку Інтернету. Як і сам Інтернет, каталог поширюється по всьому світу і зберігається на серверах доменних імен, які регулярно спілкуються один з одним для забезпечення оновлень і надмірності. Якщо в кінці 2017 року було зареєстровано більше 332 мільйонів доменних імен, то один каталог дійсно буде дуже великим.

Кожен названий сайт може відповідати більш ніж одному IP-адресою. Фактично, деякі сайти мають сотні або більше IP-адрес, які відповідають

одному доменному імені. Наприклад, сервер, до якого ваш комп'ютер підключається до www.google.com, ймовірно, повністю відрізняється від сервера, на який може потрапити хтось в іншій країні, ввівши той же ім'я сайту в своєму браузері.

Інша причина розподіленої природи каталогу - кількість часу, який знадобиться вам, щоб отримати відповідь, коли ви шукали сайт, якщо для каталогу було тільки одне місце розташування, загальна для мільйонів, ймовірно, мільярдів людей. також шукає інформацію в той же час. Це одна довга лінія, щоб використовувати телефонну книгу.

Замість цього інформація DNS розподіляється між багатьма серверами, але також кешируєтся локально на клієнтських комп'ютерах. Швидше за все, ви використовуєте google.com кілька разів в день. Замість того, щоб ваш комп'ютер кожен раз запитував у DNS-сервера ім'я IP-адреси google.com, ця інформація зберігається на вашому комп'ютері, тому йому не потрібно звертатися до DNS-сервера для дозволу імені за допомогою його IP-адреси. Додаткове кешування може відбуватися на маршрутизаторах, використовуваних для підключення клієнтів до Інтернету, а також на серверах інтернет-провайдера користувача (ISP). При такій великій кількості кешування кількість запитів, які фактично надходять на сервери DNS-імен, набагато менше, ніж може здатися.

Як DNS підвищує ефективність

DNS організований в ієрархії, яка допомагає швидко і без проблем працювати. Щоб проілюструвати це, давайте уявимо, що ви хотіли відвідати networkworld.com.

Початковий запит IP-адреси виконується для рекурсивного розпізнувача, сервера, який зазвичай управляється інтернет-провайдером або іншим стороннім постачальником. Рекурсивний розпізнувач знає, які інші DNS-сервери йому потрібно запросити для вирішення імені сайту (networkworld.com) з його IP-адресою. Цей пошук приводить до кореневого сервера, який знає всю інформацію про домени верхнього рівня, таких як .com, .net, .org і про всіх доменах таких країн, як .cn (Китай) і .uk (Великобританія). Кореневі сервери

розташовані по всьому світу, тому система зазвичай направляє вас до найближчого географічно.

Як тільки запит досягає правильного кореневого сервера, він направляється на сервер імен домену верхнього рівня (TLD), який зберігає інформацію для домену другого рівня, слова, використовувані перед тим, як ви потрапите на .com, .org, .net (наприклад, ця інформація для networkworld.com - «networkworld»). Потім запит відправляється на сервер доменних імен, який містить інформацію про сайт і його IP-адресу. Після виявлення IP-адреси він відправляється назад клієнту, який тепер може використовувати його для відвідування веб-сайту. Все це займає всього лише мілісекунди.

Оскільки DNS працює вже більше 30 років, більшість людей приймають це як належне. Безпека також не враховувалася при створенні системи, тому хакери в повній мірі скористалися цим, створюючи різні атаки.

DNS відбиття атак. Атаки на відображення DNS можуть завалити жертви великими обсягами повідомлень з серверів розпізнавання DNS. Зловмисники запитують великі файли DNS у всіх відкритих распознавателей DNS, які вони можуть знайти, і роблять це, використовуючи підроблений IP-адреса жертви. Коли розпізнавачі відповідають, жертва отримує потік незапрошених даних DNS, який перевантажує їх комп'ютери.

Отруєння кеша DNS. Отруєння кеша DNS може перенаправити користувачів на шкідливі веб-сайти. Зловмисникам вдається вставити помилкові записи адрес в DNS, тому, коли потенційна жертва запитує дозвіл адреси для одного із заражених сайтів, DNS відповідає IP-адресою для іншого сайту, який контролюється зловмисником. Потрапивши на ці фальшиві сайти, жертви можуть бути обмануті, щоб відмовитися від паролів або перенести завантаження шкідливих програм.

Вичерпання ресурсів DNS. Атаки на вичерпання ресурсів DNS можуть засмітити DNS-інфраструктуру інтернет-провайдерів, блокуючи їх доступ до сайтів в Інтернеті. Це може бути зроблено зловмисниками, реєструючими доменне ім'я і використовують сервер імен жертви в якості авторитетного сервера домена. Тому, якщо рекурсивний распознаватель не може надати IP-

адресу, пов'язану з ім'ям сайту, він запросить сервер імен жертви. Зловмисники генерують велику кількість запитів, які підтримуються вашим і завантажують неіснуючі субдомени для завантаження, що призводить до потоку запитів дозволу, що запускаються на сервері імен жертви, що призводить до його перевантаження.

Що таке DNSSec?

Розширення безпеки DNS - це спроба зробити зв'язок між різними рівнями серверів, що беруть участь в пошуку DNS, безпечнішою. Він був розроблений Інтернет-корпорацією з присвоєння імен і номерів (ICANN), організацією, що відповідає за систему DNS.

ICANN стало відомо про недоліки зв'язку між серверами каталогів верхнього рівня, другого і третього рівня DNS, які можуть дозволити зловмисникам перехоплювати пошукові запити. Це дозволило б зловмисникам відповідати на запити про пошук на законних сайтах з IP-адресою для шкідливих сайтів. Ці сайти можуть завантажувати шкідливе ПО користувачам або проводити фішингові і фарминг-атаки.

DNSSEC вирішить цю проблему, якщо кожен рівень DNS-сервера буде підписувати свої запити цифровим підписом, що гарантує, що запити, надіслані кінцевими користувачами, які не будуть зламані зловмисниками. Це створює ланцюжок довіри, так що на кожному етапі пошуку перевіряється цілісність запиту.

Крім того, DNSSec може визначити, чи існують доменні імена, і якщо немає, то не допустить, щоб шахрайський домен доставлявся невинним особам, які бажають дозволити доменне ім'я.

У міру того, як створюється все більше доменних імен, і все більше пристроїв продовжують підключатися до мережі через пристрої Інтернету та інші «розумні» системи, а також у міру того, як все більше сайтів переходять на IPv6, потрібно підтримка здорової екосистеми DNS. Зростання великих даних і аналітики також збільшує потребу в управлінні DNS.

3.4.1 Динамічний DNS

Динамічний DNS автоматично оновлює записи DNS при зміні IP-адреси. Динамічний DNS використовується в великих мережах, в яких розміщені внутрішні служби, і використовують свої власні внутрішні DNS і DHCP-сервери.

Однак невеликі компанії і домашні мережі зазвичай не мають свого власного DNS-сервера, так навіщо їм динамічний DNS?

Служби динамічного DNS використовуються невеликими компаніями і приватними особами, коли вони хочуть опублікувати службу в Інтернеті, і ця служба розміщується у внутрішній мережі, яка використовує маршрутизатор NAT для підключення до Інтернету.

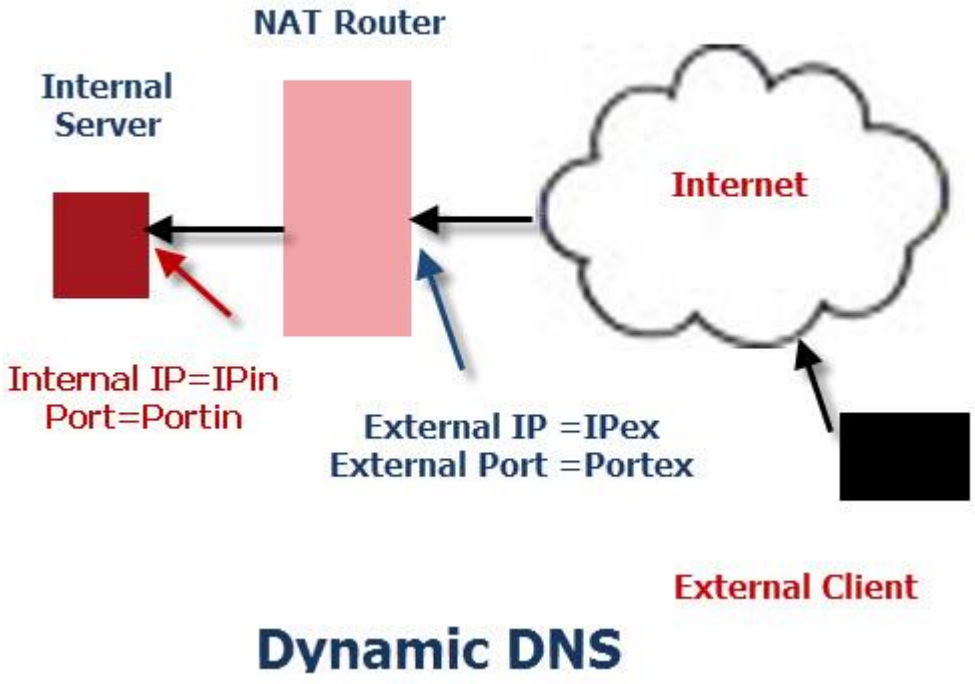

Рис.3.4 – Схема мережі нижче показує конфігурацію

Коли комп'ютери обмінюються даними через Інтернет, відправляють повідомлення один одному таким же чином, як поштові служби, і так само, як ваш місцевий листоноша, комп'ютера необхідно знати, на яку адресу надсилати вашу інформацію - це приймає форму IP-адреси і буде виглядати приблизно так: "216.239.59.99".

Для більшості людей подібні цифри означають дуже мало, і якщо б вам доводилося вводити IP-адресу для кожного веб-сайту, який ви відвідуєте, було б дуже важко використовувати, не кажучи вже про практичні проблеми запам'ятовування кожного IP-адреси, який ви відвідуєте. Для вирішення цієї проблеми була створена Система доменних імен (DNS), яка є однією з основних послуг, яка робить можливим використання Інтернету.

У спробі знизити навантаження на кореневі сервери, більшість інтернет-провайдерів будуть використовувати локальні DNS-сервери, які будуть зберігати копію IP-адрес для сайтів, які відвідують їхні клієнти. Хорошим прикладом цього є мільйони людей, які використовують Google щодня. Якби їм доводилося заходити на кореневі сервери, щоб отримувати IP-адресу для www.google.com кожен раз, коли вони хотіли виконати пошук, цей трафік міг би легко привести до відключення серверів і Інтернету разом з ним, але якщо інтернет-провайдер запускає локальний DNS-сервер, який зберігає всі IP-адреси популярних сайтів, локальний сервер може замість цього повернути результат і зробити весь сервіс швидшим і менш завантаженим для всіх.

Незважаючи на те, що DNS звучить по-справжньому чудово, його може бути досить складно налаштувати, і це принесе чимало клопоту, якщо ви просто хочете, щоб люди підключалися до вашого домашнього веб-сервера або ігрового сервера, а не йшли на неприємності, є багато компаній. це зробить це для вас безкоштовно, пропонуючи послугу, відому як «Динамічний DNS». Це працює шляхом запуску невеликої програми на вашому домашньому комп'ютері, яка буде виявляти ваш IP-адреса час від часу або навіть кожен раз, коли ви набираєте номер, а потім відправляє його в компанію, яка підтримує динамічний DNS. Як тільки ця інформація буде отримана з вашого комп'ютера, компанія буде автоматично оновлювати свої DNS-сервери на місці, щоб будь-які зміни вашого IP-адреси негайно ставали відомими будь-кому, хто намагається до вас підключитися.

Таким чином, ваше «DNS-ім'я», таке як «johndoe.dynamic-dns.com», ніколи не зміниться, але IP-адреса, на який він відображається, може

змінюватися так часто, як це потрібно, і оскільки цей процес відбувається за лаштунками, поки ви підключені до Інтернету, ваш комп'ютер завжди буде доступний кожному, хто намагається підключитися до нього, використовуючи ваше DNS-ім'я.

3.4.2 NO-IP сервіс

Vitalwerks LLC є материнською компанією No-IP, яка є динамічним постачальником DNS для платних і безкоштовних послуг. No-IP пропонує послуги DNS, електронну пошту, моніторинг мережі і SSL-сертифікати. Служби електронної пошти включають електронну пошту POP3, вихідну електронну пошту SMTP, резервні поштові служби, а також відображення і фільтрацію пошти.

No-IP був запущений в жовтні 2000 року, пропонуючи безкоштовне динамічне перенаправлення DNS і URL. Користувачі змогли створити піддомен під декількома доменами, що належать No-IP. У травні 2000 року Vitalwerks Internet Solutions, LLC була створена як материнська компанія No-IP. У січні 2001 року No-IP почав пропонувати платні керовані служби DNS, які дозволяли користувачам налаштовувати динамічний DNS, використовуючи своє власне доменне ім'я. Пізніше в тому ж році вони почали пропонувати послуги електронної пошти на додаток до свого продукту DNS. З ростом популярності No-IP з'являлася в таких журналах, як PC magazine і Mac User. Вони почали перепродувати доменні імена в 2002 році, а в 2006 році стали акредитованим реєстратором ICANN.

Основний продукт No-IP - це динамічні служби DNS («DDNS»). Базові динамічні служби DNS, що використовують домен, що належить No-IP, можна використовувати безкоштовно, поки обліковий запис залишається активною. Оновлена послуга для використання вашого власного доменного імені буде коштувати близько 25 доларів в рік. Динамічні IP-адреси є загальними для житлових кабельних або DSL-широкосмугових облікових записів. Безкоштовний сервіс дозволяє користувачам встановлювати від одного до трьох імен хостів на доменному імені, наданому No-IP. Ім'я хоста буде перетворено в

поточний IP-адреса комп'ютера користувача. No-IP для Windows, OS X і Linux також надає програмний клієнт, який можна запустити на комп'ютері з динамічним адресою. No-IP також надає інші служби DNS, електронної пошти та моніторингу мережі.

Динамічне DNS-ім'я хоста пов'язано з динамічним IP-адресою користувача. Коли IP-адреса змінюється, динамічний DNS-клієнт відправляє оновлення на No-IP з поточним IP-адресою, а потім No-IP поширює зміна DNS в Інтернет протягом декількох секунд.

Для спрощення поновлення IP-адрес No-IP має відкритий протокол, який дозволяє розробникам програмного забезпечення і виробникам устаткування зв'язуватися через HTTP, щоб повідомляти їх про зміну IP-адреси.

Багато виробників маршрутизаторів надають вбудовану підтримку динамічного протоколу DNS без IP, наприклад, Asus, D-Link, Dovado, Edimax, SonicWall, SMC і TP-Link.

Більшість сторонніх дистрибутивів вбудованого ПО, таких як DD-WRT, також включають No-IP в свій вибір постачальників DDNS для маршрутизаторів, які мають підтримку від виробника.

3.4.3 Використання NO-IP для свого проєкту

Щоб налаштувати Raspberry Pi Dynamic DNS з NOIP, мені потрібно створити безкоштовну обліковий запис на їх веб-сайті. Обов'язково запам'ятайте ім'я користувача, пароль та ім'я хоста, які ви вибрали, так як вони знадобляться нам на наступному кроці.

Після того, як я закінчу створення вашого профілю, прийшов час ввести їх в файл конфігурації DDclient. Я можу відкрити файл конфігурації, виконавши наступну команду.

```
sudo nano /etc/ddclient/ddclient.conf
```

Тепер або поновіть, або додайте наступні рядки в кінець файлу, переконавшись, що ім'я користувача, пароль та ім'я хоста замінені тими, які ви використовували для створення облікового запису No-IP.

```
protocol=dyndns2

server=dynupdate.no-ip.com

login=your_username

password=your_password

your_domain.com
```

Після того як я оновив цей файл, збережіть його і вийдіть з допомогою ctrl-X.

Тепер все, що мені потрібно зробити, це запустити ddclient, якщо ви ввели правильну інформацію на кроці 3. Все має працювати нормально. Використовуйте наступну команду, щоб перезапустити клієнт.

```
sudo /etc/init.d/ddclient restart
```

Тепер моя IP-адреса має бути оновлена, і я зможу використовувати обране доменне ім'я для підключення до вашого Raspberry Pi, якщо відкриті правильні порти.

3.4.4 Daemon в Linux

Демон - це тип програми в Unix-подібних операційних системах, який працює непомітно у фоновому режимі, а не під безпосереднім контролем користувача, чекаючи активації через виникнення певної події або умови.

Unix-подібні системи зазвичай запускають безліч демонів, в основному для задоволення запитів на послуги від інших комп'ютерів в мережі, а також для відповіді на інші програми і на апаратну діяльність. Прикладами дій або умов, які можуть ініціювати дії демонів, є конкретний час або дата, проходження заданого інтервалу часу, посадка файлу в конкретний каталог, отримання електронної пошти або веб-запит, зроблений через певну лінію зв'язку. Необов'язково, щоб виконавець дії або умови знав, що демон слухає, хоча програми часто виконують дію тільки тому, що вони знають, що неявно викликають демона.

Демони зазвичай створюються як процеси. Процес - це виконуваний (тобто працює) екземпляр програми. Процеси управляються ядром (тобто ядром операційної системи), яке присвоює кожному унікальний ідентифікаційний номер процесу (PID).

У Linux існує три основних типи процесів: інтерактивні, пакетні і демон. Інтерактивні процеси виконуються користувачем в інтерактивному режимі в командному рядку (т. Е В повнотекстовому режимі). Пакетні процеси відправляються з черги процесів і не пов'язані з командним рядком; вони добре підходять для виконання завдань, що повторюються, коли використання системи в іншому випадку низька.

Демони розпізнаються системою як будь-які процеси, батьківський процес яких має PID, що дорівнює одиниці, який завжди представляє процес init. init завжди є першим процесом, який запускається при завантаженні комп'ютера з Linux (т. е. запускається) і залишається в системі до тих пір, поки комп'ютер не буде вимкнений. init приймає будь-який процес, батьківський процес якого помирає (тобто завершується), не чекаючи статусу дочірнього процесу. Таким чином, загальний метод запуску демона включає в себе розгалуження (тобто поділ) один або два рази, і змусити батьківські (і прабатьківські) процеси померти, поки дочірній (або внучатий) процес починає виконувати свою звичайну функцію.

Деякі демони запускаються за допомогою сценаріїв ініціалізації System V, які є сценаріями (тобто короткими програмами), які запускаються автоматично при завантаженні системи. Вони можуть або виживати протягом сеансу, або регенеруватися через певні проміжки часу.

Багато демони тепер запускаються тільки в міру необхідності і одним демоном xinetd (який замінив inetd в новіших системах), а не працюють безперервно. xinetd, який називається супер-сервером TCP / IP, запускається під час завантаження і прослуховує порти, призначені процесам, перерахованим в файлі конфігурації /etc/inetd.conf або в файлі /etc/xinetd.conf. , Приклади запускаються демонів включають crond (який виконує заплановані завдання), ftpd (передача файлів), lpd (лазерний друк), rlogind (віддалений вхід в систему),

rshd (віддалене виконання команд) і telnetd (telnet).

Крім запуску операційною системою і прикладними програмами, деякі демони також можна запускати вручну. Приклади команд, що запускають демони, включають binlogd (який записує виконавчі події в зазначені файли), mysqld (сервер бази даних MySQL) і apache (веб-сервер Apache).

У багатьох Unix-подібних операційних системах, включаючи Linux, у кожного демона є один скрипт (тобто коротка програма), за допомогою якого він може бути завершений, перезапущений або перевірений його статус. Обробка цих сценаріїв заснована на рівнях виконання. Рівень виконання - це конфігурація або робочий стан системи, яке допускає існування тільки певних обраних процесів. Завантаження на іншому рівні виконання може допомогти вирішити певні проблеми, включаючи виправлення системних помилок.

Термін «демон» походить від демонів грецької міфології, які були надприродними істотами, які зараховувалися до богів і смертним і володіли особливими знаннями і сілой1. Наприклад, Сократ стверджував, що у нього був демон, який давав йому попередження і поради, але ніколи не змушував його йти за ним. Він також стверджував, що його демон демонстрував велику точність, ніж будь-яка з форм ворожіння, які практикувалися в той час.

Слово «демон» вперше було використано в комп'ютерному контексті в новаторському Project MAC (який згодом став лабораторією комп'ютерних наук MIT) з використанням IBM 7094 в 1963 році. Це використання було натхненне демоном Максвелла з фізики і термодинаміки, який був уявним агентом. це допомагало сортувати молекули різних швидкостей і працювало невтомно в фоновому режимі. Потім цей термін використовувався для опису фонових процесів, які працювали не покладаючи рук для виконання системних обов'язків

Першим комп'ютерним демоном була програма, яка автоматично робила резервні копії на стрічку. Після того, як термін був прийнятий для використання в комп'ютері, він був раціоналізовано як абревіатура для Disk And Execution MONitor.

В операційних системах Microsoft Windows програми, звані службами, виконують функції демонів, хоча термін «демон» тепер іноді використовується і щодо цих систем.

3.4.5 ddclient утиліта

DDclient - це клієнт Perl, який використовується для оновлення записів динамічного DNS для облікових записів в постачальника мережевих послуг динамічного DNS. Спочатку він був написаний Полом Беррі і в даний час в основному від Wimpunk. Він має можливість оновлювати більше, ніж просто dyndns, і він може отримати ваш WAN-ipaddress декількома різними способами. Перевірте сторінки конфігурації, щоб знайти, як це зробити.

3.4.6 Запуск ddclient як Daemon

Оскільки ми не хочемо, щоб IP-адреса оновлювався один раз, нам все одно потрібно налаштувати ddclient для запуску в якості демона, щоб він міг періодично перевіряти зміна IP-адреси і при необхідності повідомляти обраного динамічного DNS-провайдера.

Щоб DDclient працював як демон, нам потрібно внести зміни в файл конфігурації. Виконуємо наступну команду, щоб почати редагування файлу:

```
sudo nano /etc/default/ddclient
```

У цьому файлі вам потрібно змінити run_daemon, щоб він відповідав рядку нижче. Ця зміна в основному говорить про запуск ddclient в режимі демона.

```
run_daemon="true"
```

Вам також необхідно перевірити дві інші рядки в цьому файлі, щоб переконатися, що вони помилкові. В іншому випадку ddclient НЕ буде коректно працювати в режимі демона. Переконайтеся, що наступні два рядки співпадають, якщо для них встановлено значення true, змініть їх на false.

Натисніть Ctrl + X, а потім Y, щоб зберегти зміни у файлі конфігурації.

Тепер введіть наступну команду для запуску ddclient як служби.

```
sudo service ddclient start
```

Щоб переконатися, що служба правильно запущена, ви можете використовувати наступну команду для перевірки її стану.

```
sudo service ddclient status
```

Деякі постачальники динамічних DNS вимагають, щоб ви оновлювали свій IP-адресу досить часто. Ми можемо допомогти вам уникнути тайм-ауту, змушуючи ddclient оновлювати ваш IP-адресу раз в тиждень. Ми досягаємо цього, спочатку відкриваючи щотижневий crontab, використовуючи наступну команду.

```
sudo nano /etc/cron.weekly/ddclient
```

До цього файлу додайте наступні рядки:

```
#!/bin/sh

/usr/sbin/ddclient -force
```

Збережіть зміни, натиснувши Ctrl-X, а потім Y, щоб прийняти зміни.

Нарешті, нам потрібно дозволити виконання нашого нового скрипта, і ми можемо зробити це швидко, виконавши такі дії.

```
sudo chmod +x /etc/cron.weekly/ddclient
```

Тепер ddclient повинен успішно працювати в якості демона, використовуйте наступну команду, щоб переконатися, що він працює правильно:

```
sudo service ddclient status
```

Це повинно бути все, що вам потрібно зробити, щоб все працювало гладко.

Якщо ви виявите, що щось не працює правильно, то наступний рядок відмінно підходить для налагодження. Він буде плювати тонну рядків, просто шукати все, що може виглядати як помилка або натяк на те, чому це не працює.

```
sudo ddclient -daemon=0 -debug -verbose -noquiet
```

Як тільки у вас встановлена переадресація портів Raspberry Pi, тепер ви зможете отримати доступ до вашого Pi поза локальної мережі, використовуючи доменне ім'я. Якщо наведені вище інструкції не працюють для вас, ви завжди можете спробувати старий метод, який детально описаний нижче.

3.5 Останні кроки для налаштування

Введіть друге, щоб відкрити наш файл сервера за замовчуванням:

```
sudo nano /etc/nginx/sites-available/default
```

Після поновлення IP-адрес у файлі сервера вам потрібно буде додати зовнішній IP-адреса в список довірених IP-адрес і переконатися, що Owncloud також не буде перезаписано його. Для цього відкрийте конфігураційний файл Owncloud і введіть:

```
sudo nano /var/www/owncloud/config/config.php
```

Тут додайте новий елемент в масив довірених доменів (це буде ваш зовнішній IP-адреса). Ваша нова запис має виглядати приблизно так (x - просто наповнювачі).

```
1 => 'xxx.xxx.xxx.xxx',
```

Нарешті, поновіть URL-адресу рядки overwrite.cli.url до вашого IP-адреси. Це має виглядати приблизно так:

```
'overwrite.cli.url' => 'https://xxx.xxx.xxx.xxx',
```

Після цього перезапустіть службу Nginx, ввівши наступне:

ВИСНОВОК ДО РОЗДІЛУ 3

В даному розділу я постарався розповісти докладно і зрозуміло про повне налаштування власного хмарного сховища за допомогою Rasbperry Pi. Більшість матеріалів було взято з офіційних документацій перелічених у проєкті технологій. В розділі було детально розглянуто технологія DNS, Daemon сервіс.

Особливу я приділив сервісу ownCloud, який ми використовуємо як клієнт для хмарного сховища. Він підтримується великою кількістю людей з усього світу, так як є проєктом відкритом кодом.

Ми змогли налаштувати нашу хмару таким шляхом, що можна мати до неї доступ з будь-якої точки світу. Це все було зроблено за допомогою сервісу NO-IP.

ВИСНОВОК

У данній дипломній роботі був проведений аналіз хмарних технологій, зокрема хмарних сховищ. Також, проведено огляд сучасного направлення IoT в техно-світі. Ми познайомилися з надзвичайним пристроєм Raspberry Pi, навчилися користуватися Linux та його терміналом. Особливо мені сподобалося налаштовувати конфігурації серверної частини, щоб усе працювало коректно.

Підводячи підсумок скажу, що з цією системою потрібно дуже вдумливо і уважно розбиратися. Але, результат вражатиме. Адже, у вас буде власне хмарне сховище розміром з кредитну картку. Ви сковані ніякими обмеженнями та можете налаштовувати свій сервіс для себе та під себе. Також, мене вразила наявність хмарного клієнту для абсолютно усіх платформ, від ПК до смартфоні. Якщо вам щось приносить задоволення, то робіть це і ніколи не зупиняйтесь. Адже, після того, як у вас вийде, ви отримаєте неймовірні відчуття.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. База проектів для Rasbperry Pi - https://pimylifeup.com/
2. Основний пошук ресурсів - https://pimylifeup.com/
3. Допоміжний пошуковий ресурс - https://duckduckgo.com/
4. Nginx - https://nginx.org/en/docs/
5. Raspberry Pi офіційний ресурс- https://nginx.org/en/docs/
6. Завантаження NOOBS - https://www.raspberrypi.org/downloads/
7. Гайд зі встановлення операційної системи NOOBS - https://www.raspberrypi.org/downloads/
8. Команди для терміналу - https://help.ubuntu.com/community/UsingTheTerminal
9. ownCloud документація - https://help.ubuntu.com/community/UsingTheTerminal